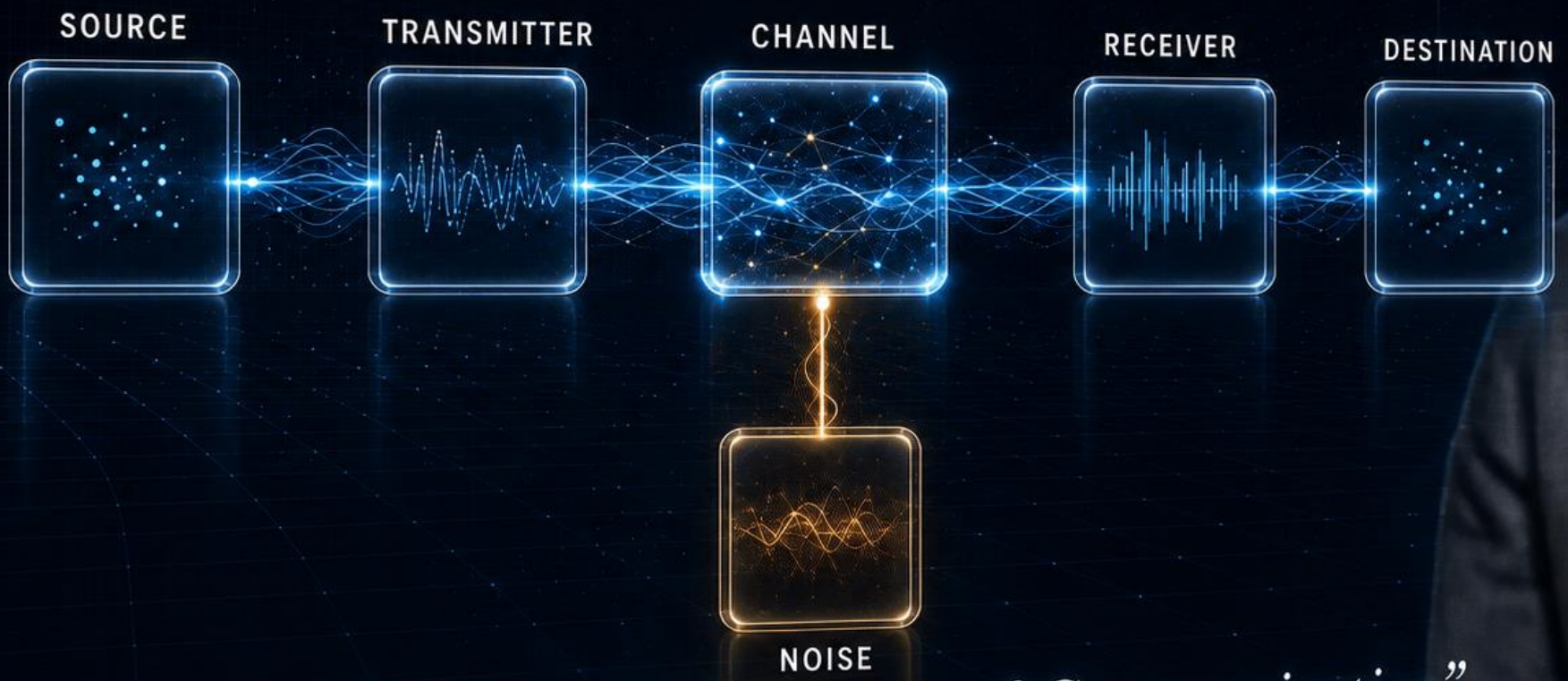
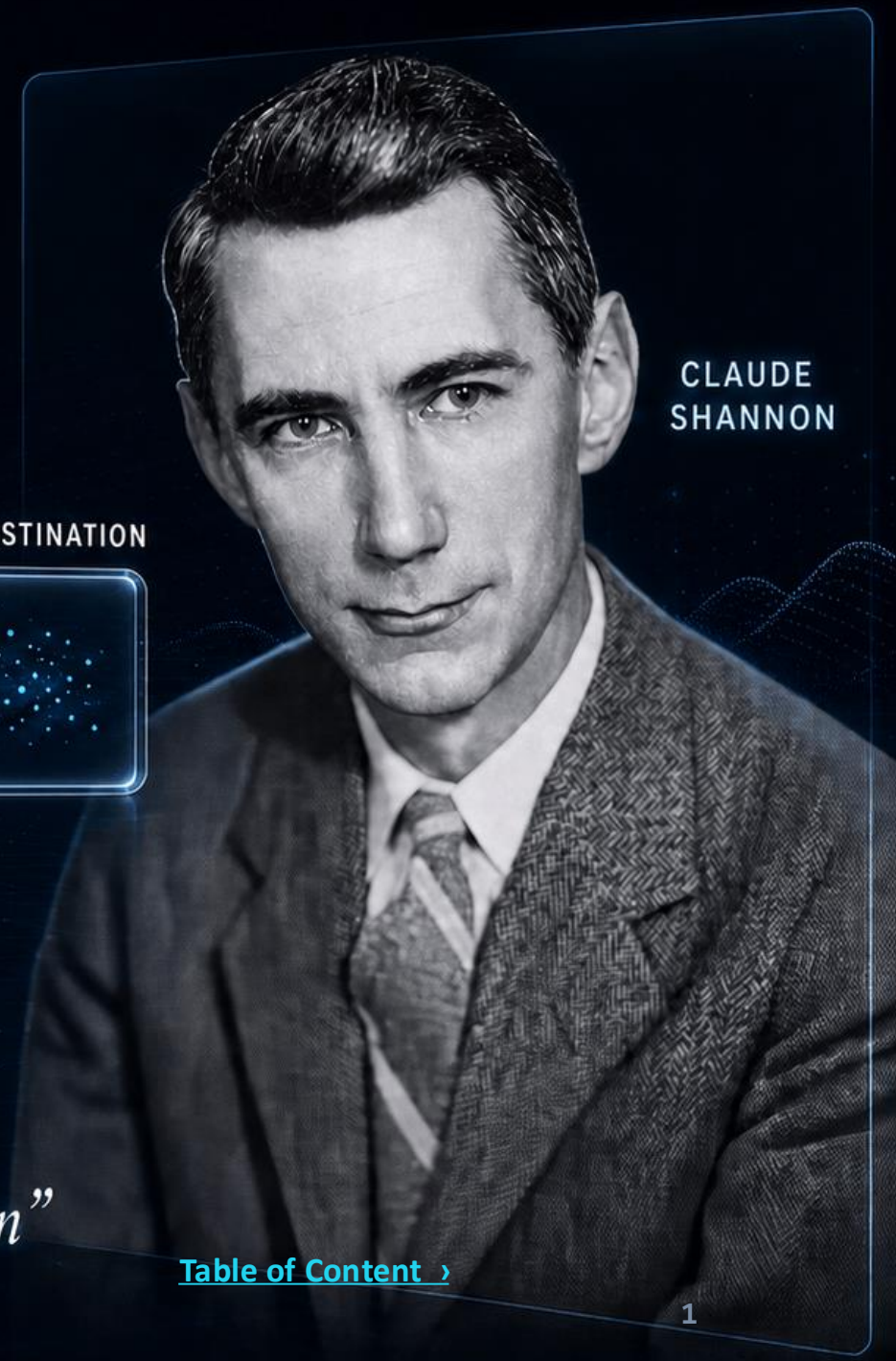
ELEC 3203 DIGITAL CODING & TRANSMISSION

SCHOOL OF ELECTRONICS AND COMPUTER SCIENCE
UNIVERSITY OF SOUTHAMPTON

WINTEL
Wireless Intelligence & Networked Sensing Laboratory

Prof. Abdulkadir Çelik
https://akadircelik.com





"The Mathematical Theory of Communication"

Table of Content ›

Course contents

Information Theory Fundamentals · seven lectures, slide numbers at right

Lecture Notes & Course Page
akadircelik.com/teaching/elec3203/

ELEC3203 · DIGITAL CODING & TRANSMISSION

LECTURE 1

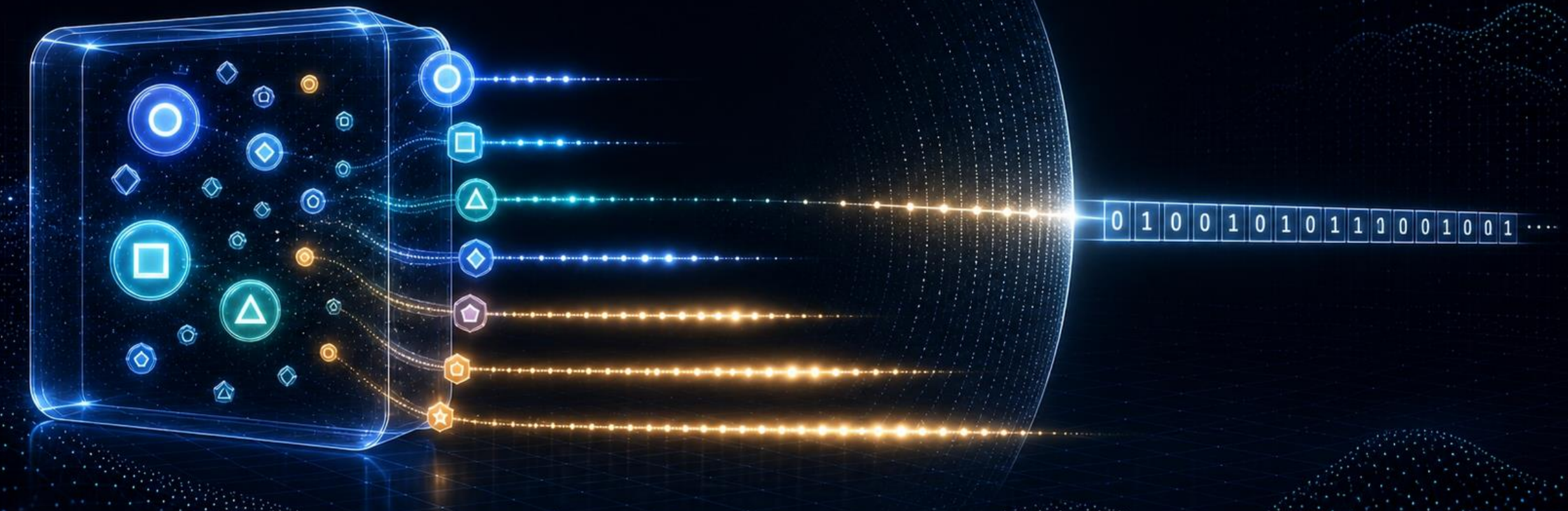
INFORMATION, ENTROPY AND THE DIGITAL SOURCE

WINTEL

Wireless Intelligence & Networked Sensing Laboratory

Prof. Abdulkadir Çelik

https://akadircelik.com



Lecture Notes

Podcast

Table of Content

Roadmap

Information · entropy · information rate · source coding

01 **A primer on the digital communication system**

Tx, channel, Rx — resources spent and KPIs targeted

02 **What is information?**

Surprise as a measurable quantity

03 **Information content of a symbol**

I(mi) = −log2 pi, and why it must be a logarithm

04 **Entropy**

Average information per symbol, H

05 **Information rate**

R = Rs · H, the true demand of a source

06 **Why BCD wastes bandwidth**

The gap between Rb and R, and coding efficiency

NOTATION

Symbols used in this lecture

m_i
the i-th symbol in the source alphabet

q
number of symbols in the alphabet

p_i
probability that symbol m_i occurs, $\sum p_i = 1$

R_s
symbol rate of the source (symbols/second)

$I(m_i)$
self-information of symbol m_i, in bits

H
entropy: average information per symbol (bits/symbol)

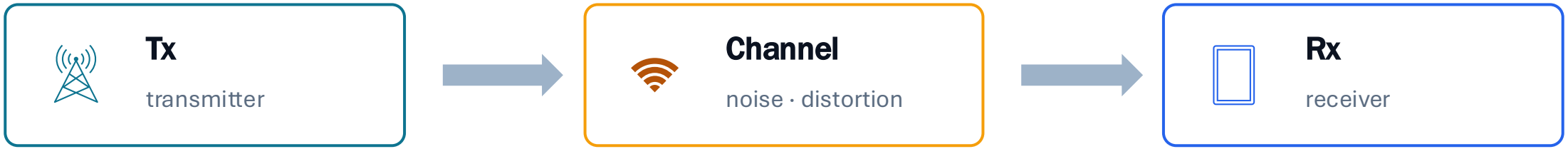
R
information rate, $R = R_s \cdot H$ (bits/second)

R^b
output bit rate of a code; $R_s \cdot \log_2 q$ for BCD

L
average codeword length (bits/symbol)

η
coding efficiency, $\eta = H / L$

The whole module in miniature



Convey information between two locations, over a channel of adequate quality, at a certain rate, reliably.

RESOURCES SPENT

- Power
- Time
- Bandwidth
- Space

KPIs TARGETED

- Rate
- Latency
- Reliability
- Energy eff.

Every technique in this module — source coding, channel coding, modulation — is a way of trading these resources for these KPIs. Lecture 1 works on the source end: how many bits per second does the source truly need?

ELEC 3203 · DIGITAL CODING & TRANSMISSION

CELLULAR NETWORK EVOLUTION

2G → 3G → 4G → 5G → 6G

WINTEL
Wireless Intelligence & Networked Sensing Laboratory
Prof. Abdulkadir Çelik
https://akadircelik.com

ELEC 3203 · DIGITAL CODING & TRANSMISSION

WIRELESS CONNECTIVITY BY SCALE

PAN → WLAN → MAN → LPWAN / WAN

WINTEL
Wireless Intelligence & Networked Sensing Laboratory
Prof. Abdulkadir Çelik
https://akadircelik.com

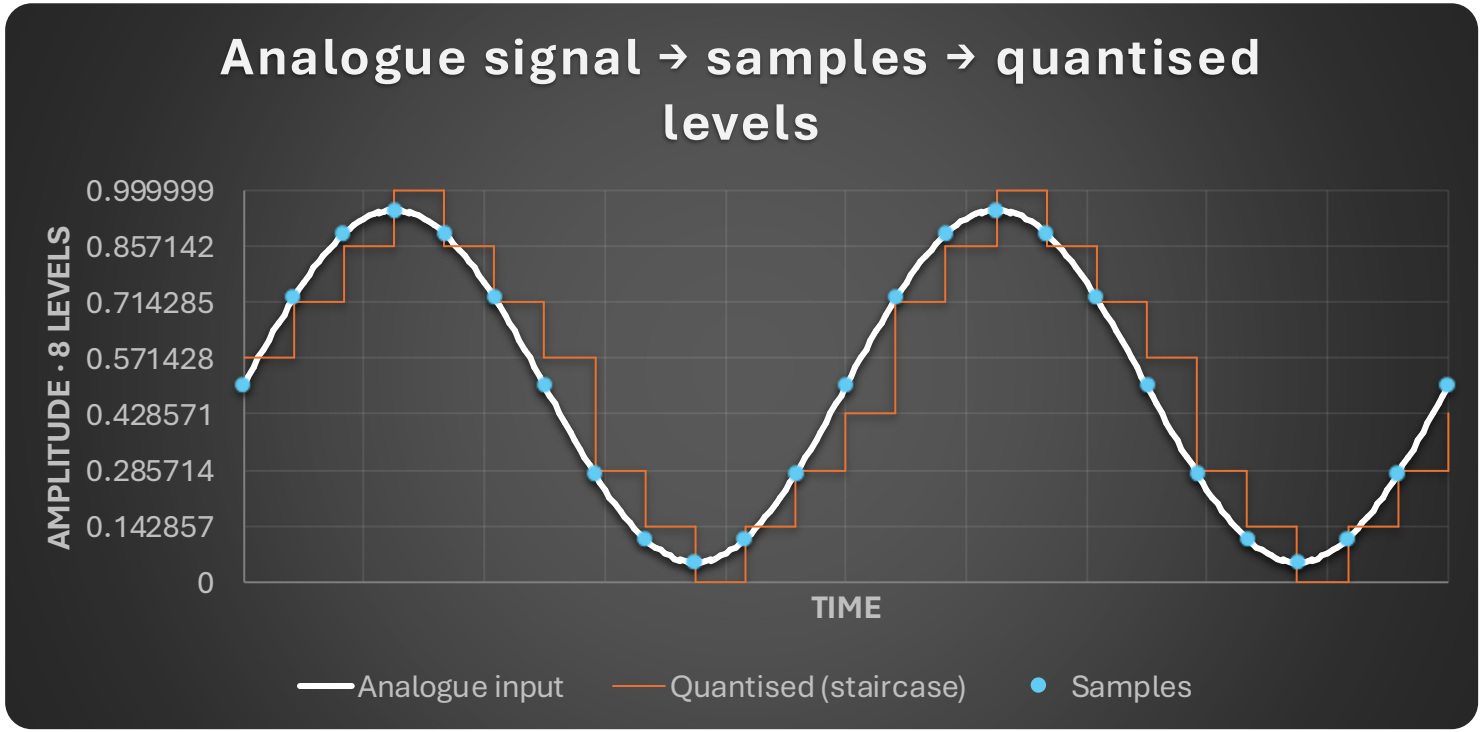
Sampling, quantisation, encoding

1 Analogue input
Continuous, not discrete

2 Sampling
Nyquist: at least 2 × f_{max}

3 Quantisation
Snap to a finite set of levels

4 Bit stream
One code word per level



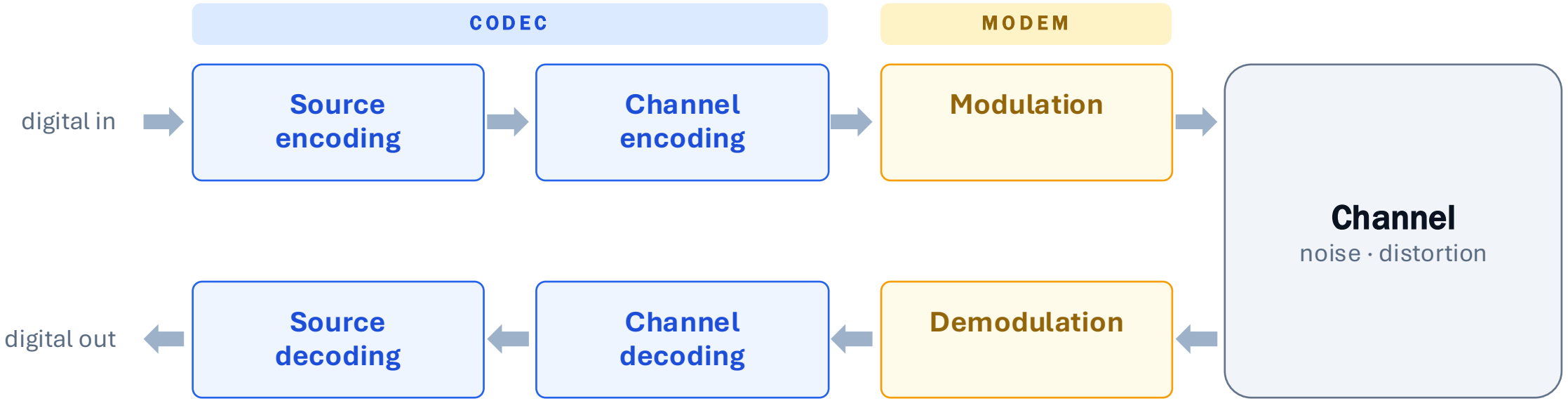
ENCODED BIT STREAM

```
010 011 100 011 010 001 000
001 010
```

Sampling is lossless if the Nyquist rule is met.

Quantisation is lossy — rounding adds quantisation noise.

CODEC, MODEM and the channel



Transmitter along the top, receiver along the bottom; a CODEC and MODEM pair on each side together forms a transceiver.

VERTICAL GROUPING

Splits the chain into three jobs: the CODEC does source and channel coding, the MODEM does modulation and demodulation, and the channel is the medium.

HORIZONTAL GROUPING

Separates transmitter (top) from receiver (bottom). Each side pairs a CODEC with a MODEM to form a transceiver — the radio inside your phone.

Small, then robust, then physical

Source coding

Squeezes the message down by removing redundancy, so it uses as few bits as possible — ideally close to the entropy H.

Huffman · ZIP · JPEG · MP3

Channel coding

Adds a controlled amount of redundancy back in, so the receiver can detect and correct the errors noise introduces. It deliberately works against source coding.

Parity · Hamming · Turbo · LDPC

Modulation and demodulation

Maps the bit stream onto a physical waveform that suits the medium — a voltage level, a radio carrier, a light pulse. The demodulator recovers the bits at the far end.

ASK · FSK · PSK · QAM · OFDM

Channel

The physical medium the waveform travels through. It is never perfect: it weakens the signal and adds noise, and that is what ultimately limits the rate.

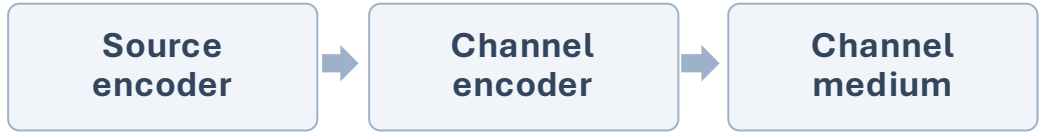
Copper · Optical fibre · Free space

Source coding makes the stream small · channel coding makes it robust · modulation makes it physical. The receiver undoes each step in reverse.

LOOKING AHEAD

Joint source-channel coding, and where AI enters

CLASSICAL · SEPARATED DESIGN



Two independently optimised blocks. Shannon's separation theorem says this loses nothing — but only for a point-to-point link with ideal, infinite-length codes.

JSCC · JOINT, OFTEN LEARNED

One stage designed — or learned — end to end, mapping source symbols straight to channel symbols. Increasingly the standard in modern wireless.

Why separation breaks

Real transceivers have neither infinite code length nor a fixed, known channel, so separated designs fail abruptly when quality drops.

Deep-learning JSCC

Autoencoder-style neural encoders and decoders learn the source-to-channel mapping from data, degrading gracefully instead of hitting the cliff effect.

Towards 6G

This underpins the semantic and task-oriented communication schemes now being explored for sixth-generation systems.

Lecture 1 stays on the classical path — but this is where the field is heading.

THE DESTINATION

Shannon's theorem

If the information rate of a source does not exceed the capacity of a channel, then there exists a coding scheme that transmits the information over that channel with an arbitrarily small probability of error, despite the noise. If the rate exceeds capacity, no coding scheme can save you.

$$R \leq C \implies \text{reliable transmission is possible}$$

An existence result

It promises that a code exists once $R \leq C$, but it does not tell us how to build one.

Silent on cost

It says nothing about delay or complexity. Reaching capacity in practice took until turbo codes in the 1990s.

Capacity comes later

Lecture 6 defines C. For bandwidth B and signal-to-noise ratio S/N it takes the Shannon-Hartley form $C = B \log_2(1 + S/N)$.

Lecture 1 works on the left-hand side: measuring the information rate R that a source produces.

1.2

PART TWO

Measuring information

Surprise as a measurable quantity

Information content | Entropy | Information rate | BCD waste

The same three words, “it will rain”

Rainforest, wet season

$p \approx 1$

Rain is almost certain, so its announcement tells us something we already knew.

Information carried: LOW

Somewhere in England

$p \approx 0.5$

Genuinely uncertain, so the answer carries a moderate amount of information.

Information carried: MODERATE

Desert

$p \approx 0$

Rain is very rare, so the announcement would be a genuine shock.

Information carried: HIGH

THE CENTRAL INTUITION

Information is tied to uncertainty. The more unexpected an event is — the smaller its probability — the more information its occurrence conveys. That single idea is the seed of the entire theory.

Four quantities describe any source

1

Alphabet

The set of possible messages.

$\{ m_1, m_2, \ldots, m_q \}$

2

Symbol probabilities

How often each symbol occurs.

$p_1, \ldots, p_q \quad \sum p_i = 1$

3

Symbol rate

How fast symbols are emitted.

$R_s \quad [\text{symbols/s}]$

4

Memory

Statistical dependence between symbols.

$p(m_k \mid m_{k-1})$

A memoryless source has symbols that are independent of one another — that is the case we study in this lecture. Sources with memory are the subject of Lecture 3.

Does the past help predict the next symbol?

INDEPENDENT (MEMORYLESS)

Fair coin tosses

Each toss is unaffected by the last

Lottery draw with replacement

The machine has no memory of past balls

Rolls of a die

Six outcomes, always equally likely

Knowing the last symbol tells you nothing about the next one, so there is no redundancy to remove. This is the case treated in Lecture 1.

CORRELATED (WITH MEMORY)

English text

A “q” is nearly always followed by a “u”

Consecutive video frames

Successive frames are almost identical

Daily temperature

Today predicts tomorrow rather well

Here the past helps predict the next symbol, so a coder can exploit that redundancy. Markov models for such sources are Lecture 3.

Three requirements pin down the function

1 Monotone in surprise

A smaller probability p_i must give a larger I.

2 Certainty carries nothing

We need $I = 0$ when $p_i = 1$.

3 Additive for independence

Two independent symbols together, with probability $p_i p_j$, must carry $I(m_i) + I(m_j)$.

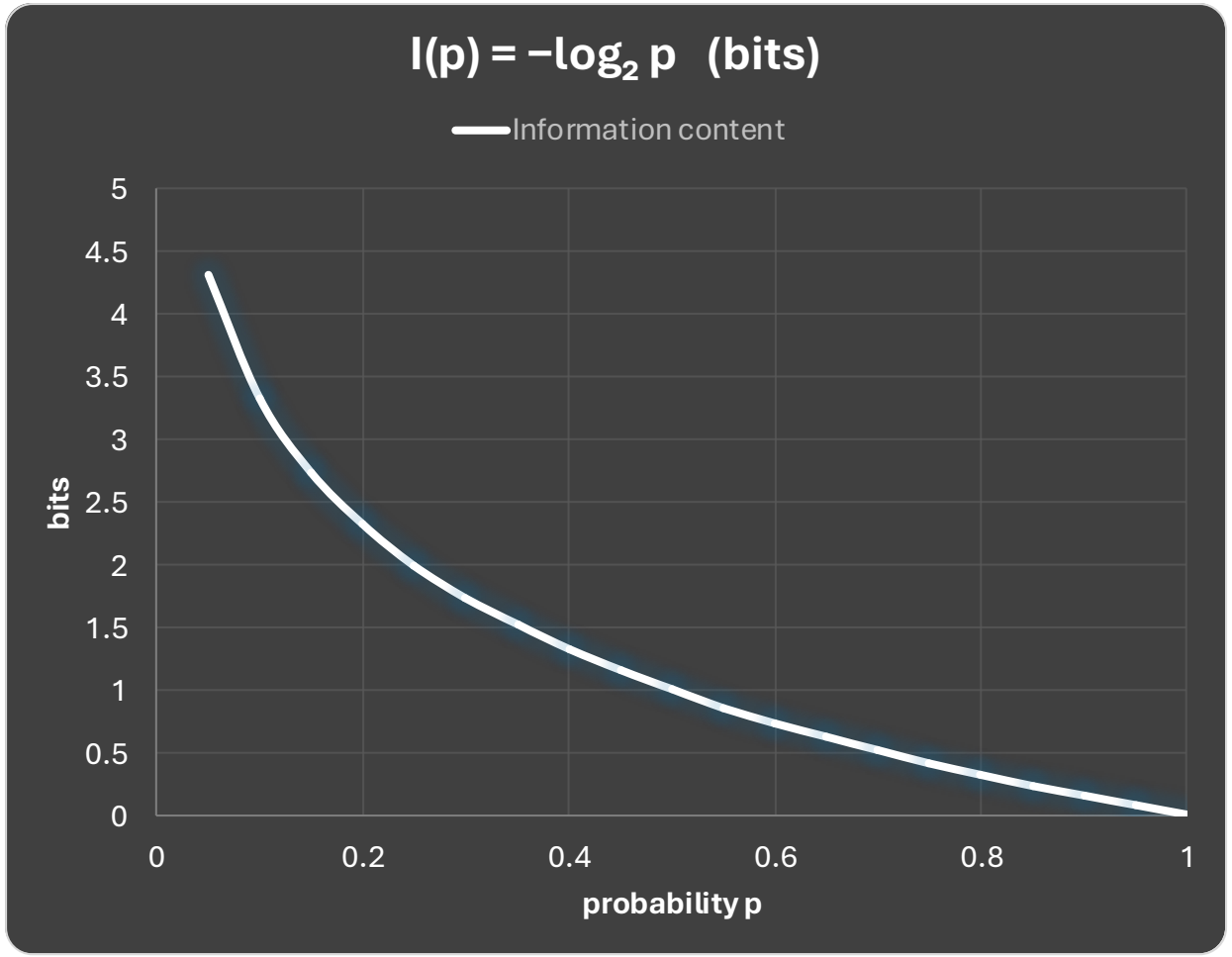
THE DECISIVE REQUIREMENT

$I(p_i\, p_j) = I(p_i) + I(p_j)$ — a function that turns products into sums. Only the logarithm does that, so $I(p) \propto \log(1/p)$.

$$I(m_i) = \log_2 (1 / p_i) = -\log_2 p_i \qquad \text{(bits)}$$

Shannon (1948, Bell Labs) chose base 2 because it matches the natural unit of a digital system — the bit, a name he credited to John Tukey. Base e gives nats, base 10 gives hartleys; only the ruler changes.

The shape of $I(p) = -\log_2 p$



PROPERTIES

$I(m_i) \geq 0$

Since $0 \leq p_i \leq 1$, information is never negative.

$p_i > p_j \Rightarrow I(m_i) < I(m_j)$

The rarer the symbol, the more it tells you.

$I \to 0$ as $p_i \to 1$

A certain symbol carries no information at all.

$I \to \infty$ as $p_i \to 0$

A vanishingly rare symbol carries unbounded information.

Why the unit is called the bit

For equally likely symbols, the information content equals the smallest number of binary digits needed to label a symbol.

q = 2

{ 0, 1 }

one binary digit each

$I = \log_2 2 = 1$ bit

q = 4

{ 00, 01, 10, 11 }

two binary digits each

$I = \log_2 4 = 2$ bits

q symbols

{ … q labels … }

$\log_2 q$ binary digits each

$I = \log_2 q$ bits

BINARY CODED DECIMAL (BCD)

Assigning $\log_2 q$ bits to every symbol is BCD. It is exactly right when symbols are equally likely.

When symbols are not equally likely, BCD wastes bits — and fixing that waste is what Lectures 1 to 4 are about.

Average information per symbol

1

A long sequence

Take N symbols from a memoryless source. Symbol m_i appears about

$p_i N$ occurrences

2

One symbol's share

Each occurrence is worth $I(m_i)$, so m_i contributes

$I_i = (p_i N) \log_2(1/p_i)$

3

Total information

Sum over the whole alphabet.

$I = \sum p_i N \log_2(1/p_i)$

4

Divide by N

N cancels, leaving the average per symbol — the entropy H.

$H = I / N$

$$H = \sum p_i \log_2 (1/p_i) = -\sum p_i \log_2 p_i \quad \text{(bits/symbol)}$$

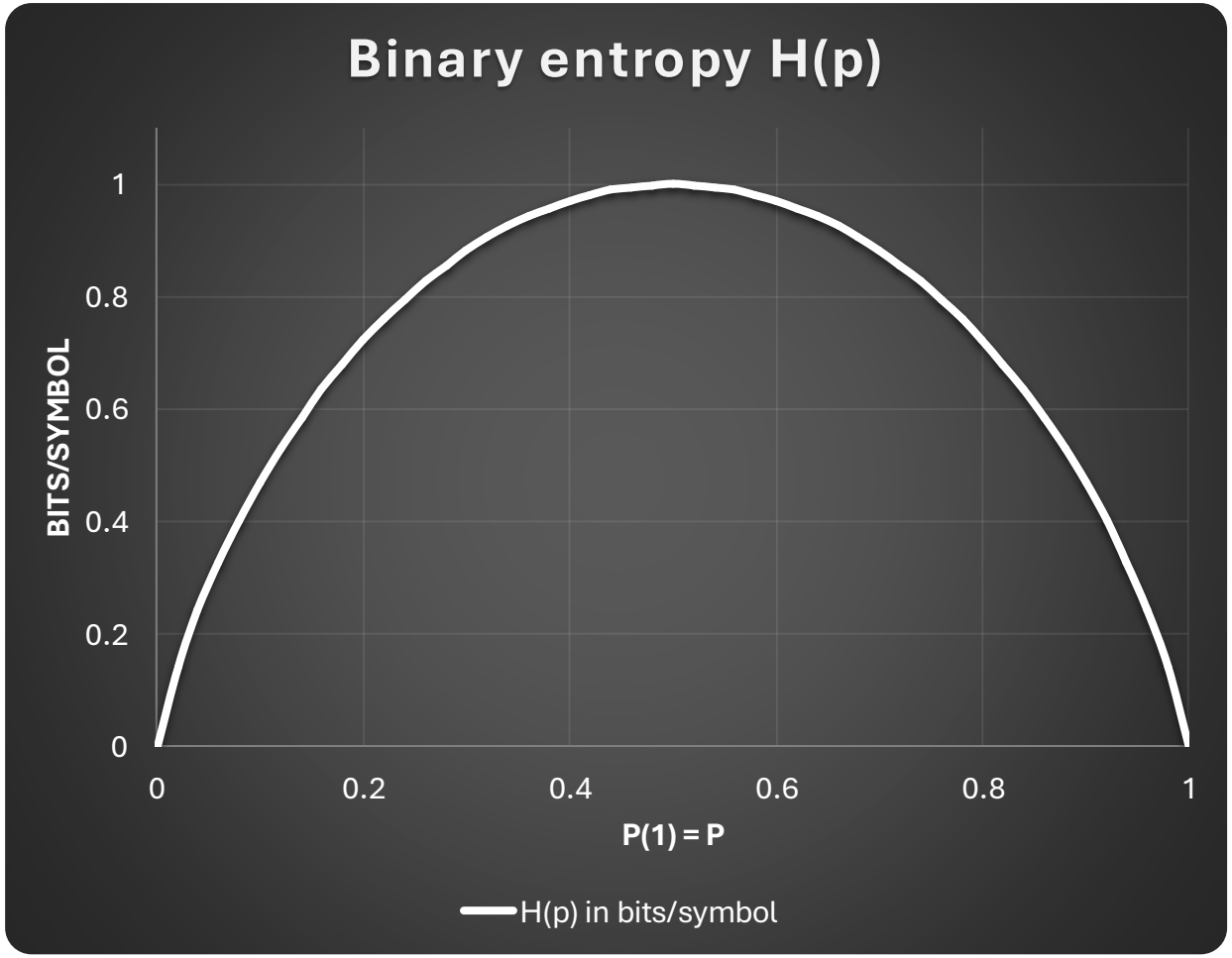
A PROPERTY OF THE SOURCE

H depends only on the alphabet and the probabilities — not on the symbol rate.

CONVENTION: 0 log 0 = 0

A symbol with $p_i = 0$ never occurs, and $p \log_2 p \to 0$, so it contributes nothing.

Maximum entropy is the fair coin



Zero at both ends

A coin that almost always lands the same way is predictable, so it carries almost no information.

Maximum at p = 0.5

Exactly 1 bit at the fair coin — the point of greatest uncertainty.

GENERALISING TO q SYMBOLS

Entropy is greatest when all symbols are equally likely, $p_i = 1/q$, giving $H_{max} = \log_2 q$.

$$0 \leq H \leq \log_2 q$$

From bits per symbol to bits per second

H
Entropy
bits / symbol

×

R_s
Symbol rate
symbols / second

=

R
Information rate
bits / second

$$R = R_s \cdot H \quad \text{(bits/second)}$$

Depends on all four quantities

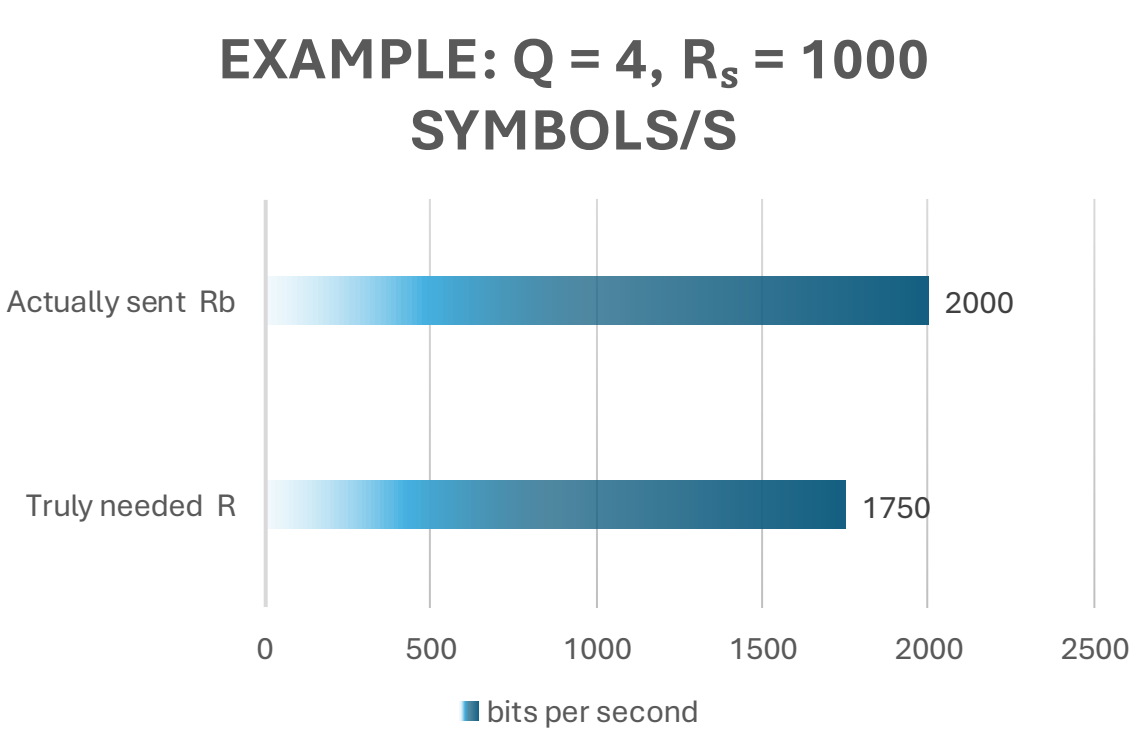
Unlike entropy, the information rate includes the symbol rate R_s — so it reflects the whole source, not just its statistics.

The target to hit

R is the true, irreducible demand of the source in bits per second. Any good encoder tries to get as close to it as possible.

Example: $H = 1.75$ bits/symbol and $R_s = 1000$ symbols/s → $R = 1750$ bits/s

Sending more bits than you need



A 12.5% waste — 250 bits per second of pure overhead.

BCD output bit rate

$$R^b = R_s \cdot \log_2 q$$

Because $H \le \log_2 q$

$$R = R_s H \le R_s \log_2 q = R^b$$

The gap is pure waste

$R^b - R$ bits/second of nothing

Coding efficiency $\eta = H / L$ — BCD gives $L = \log_2 q$, so $\eta = H / \log_2 q \le 1$. A perfect entropy code reaches $\eta = 1$.

1.7

PART THREE

Worked examples

Four exam-style problems, fully solved

- 1.1 Information content
- 1.2 Entropy and BCD waste
- 1.3 Biased binary source
- 1.4 English text

WORKED EXAMPLES 1.1 AND 1.2

A four-symbol source

Problem: $p_1 = 1/2$, $p_2 = 1/4$, $p_3 = 1/8$, $p_4 = 1/8$, emitted at $R_s = 1000$ symbols/s.

Symbol	p_i	$I(m_i)$ bits	$p_i \cdot I(m_i)$
m_1	1/2	1	0.500
m_2	1/4	2	0.500
m_3	1/8	3	0.375
m_4	1/8	3	0.375
Σ	1	—	1.750

Entropy	$H = 1.75$ bits/symbol
Information rate	$R = R_s H = 1750$ bits/s
BCD bit rate	$R^b = 2000$ bits/s
Efficiency	$\eta = 1.75 / 2 = 87.5\ \%$

READING THE RESULT

BCD sends 2000 bits per second but only 1750 are needed — a 12.5 % waste. An ideal code using 1, 2, 3 and 3-bit words would send exactly 1750 bits per second.

Bias and real language

EXAMPLE 1.3

A binary source has P(1) = 0.1 and P(0) = 0.9. Compute its entropy and compare it with a fair binary source.

$$H = -0.1 \log_2 0.1 - 0.9 \log_2 0.9$$
$$= 0.1(3.322) + 0.9(0.152)$$
$$= 0.332 + 0.137$$

$H \approx 0.47$ bits/symbol

Less than half the 1 bit of a fair coin. Because the stream is so predictable, a good code needs well under one bit per symbol — yet BCD stubbornly spends a full bit on each.

EXAMPLE 1.4

Treat English text as a 27-symbol memoryless source — 26 letters plus space, all equally likely. What does that predict, and how does it compare with reality?

$$H = \log_2 27$$
$$= 4.75 \text{ bits/letter}$$

real frequencies → 4.0 – 4.2 bits

4.75 vs ≈ 4.1 bits/letter

Letters are not equally likely: “e” and “t” are common, “z” and “q” are rare. Accounting for that alone removes over half a bit per letter — and memory removes more still.

SUMMARY

Key formulas from Lecture 1

Information content

$$I(m_i) = -\log_2 p_i$$

bits

Entropy

$$H = -\sum p_i \log_2 p_i$$

bits / symbol

Maximum entropy

$$H \leq \log_2 q$$

equality when $p_i = 1/q$

Information rate

$$R = R_s \cdot H$$

bits / second

BCD bit rate

$$R^b = R_s \log_2 q$$

always ≥ R

Coding efficiency

$$\eta = H / L$$

≤ 1, equals 1 when L = H

NEXT: LECTURE 2 — EFFICIENT SOURCE CODING

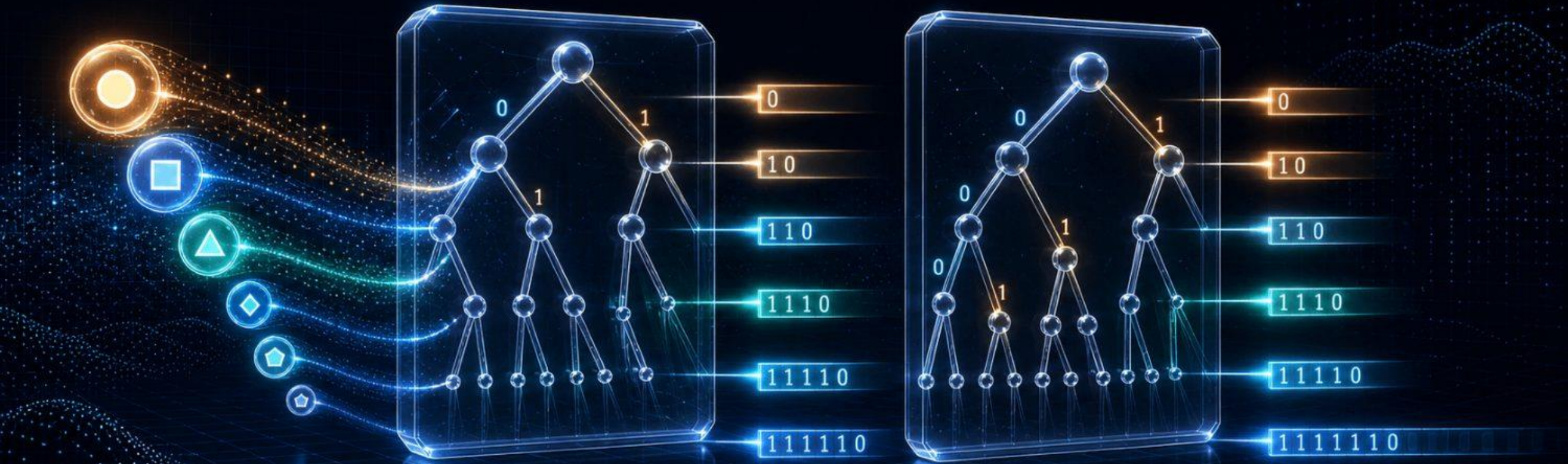
Shannon-Fano and Huffman coding: variable-length codes that push the average codeword length L down towards the entropy bound, closing the gap that BCD leaves open.

ELEC3203 · DIGITAL CODING & TRANSMISSION

LECTURE 2

EFFICIENT SOURCE CODING SHANNON-FANO & HUFFMAN

WINTEL
Wireless Intelligence & Networked Sensing Laboratory
Prof. Abdulkadir Çelik
https://akadircelik.com



Lecture Notes
Podcast
Table of Content

Roadmap

Maximum entropy · coding efficiency · Shannon-Fano · Huffman

01 **Maximum entropy**

q-ary and binary sources, derived with Lagrange multipliers

02 **Efficient source coding**

Coding efficiency CE = H / L and the design rule

03 **Shannon-Fano coding**

Top-down splitting of the sorted symbol list

04 **Huffman coding**

Bottom-up merging of the least probable symbols

05 **Comparing the two**

Optimality, prefix codes, and practical limits

NOTATION

New symbols in this lecture

H	source entropy in bits/symbol, from Lecture 1
ℓ_i	codeword length assigned to symbol m_i, in bits
L	average codeword length, $L = \sum p_i \ell_i$
$CE = \eta$	coding efficiency, $\eta = H / L$
prefix code	no codeword is the start of another, so decoding is unambiguous
λ	Lagrange multiplier enforcing $\sum p_i = 1$

Where does $H \le \log_2 q$ come from?

1 Form the Lagrangian

Objective plus λ × the constraint. The penalty vanishes where the constraint holds, so L = H.

2 Set every partial to zero

At the top of a smooth hill the slope is flat in every direction, so $\partial L/\partial p_i = 0$ for each i, and $\partial L/\partial \lambda = 0$.

3 Solve the system

One equation per variable pins down the maximising distribution exactly.

MAXIMISE H SUBJECT TO $\sum p_i = 1$

$$L = \sum (-p_i \log_2 p_i) + \lambda (1 - \sum p_i)$$

$$\partial L/\partial p_i = -\log_2 p_i - \log_2 e - \lambda = 0$$

$$\Rightarrow \log_2 p_i = -(\log_2 e + \lambda) \quad - \text{ the same for every i, so } p_i = c$$

$$\sum c = q\,c = 1 \quad \Rightarrow \quad c = 1/q$$

using $d/dx\,(x \log_2 x) = \log_2 x + \log_2 e$

$$H_{max} = \log_2 q \quad \text{at} \quad p_i = 1/q$$

Coding efficiency and the design rule

CODING EFFICIENCY

$$CE = \eta = \text{source information rate} / \text{average output rate} = R_s H / R_s L = H / L$$

THE DESIGN RULE

Give each symbol a codeword whose length is as close as possible to its own information content. Common symbols get short codewords; rare symbols get long ones.

$$\ell_i \approx I(m_i) = -\log_2 p_i$$

Whole numbers of bits

Codeword lengths must be integers, so ℓ_i can only approximate $-\log_2 p_i$.

Prefix-free

No codeword may be the start of another, so a decoder can tell where each one ends by reading left to right.

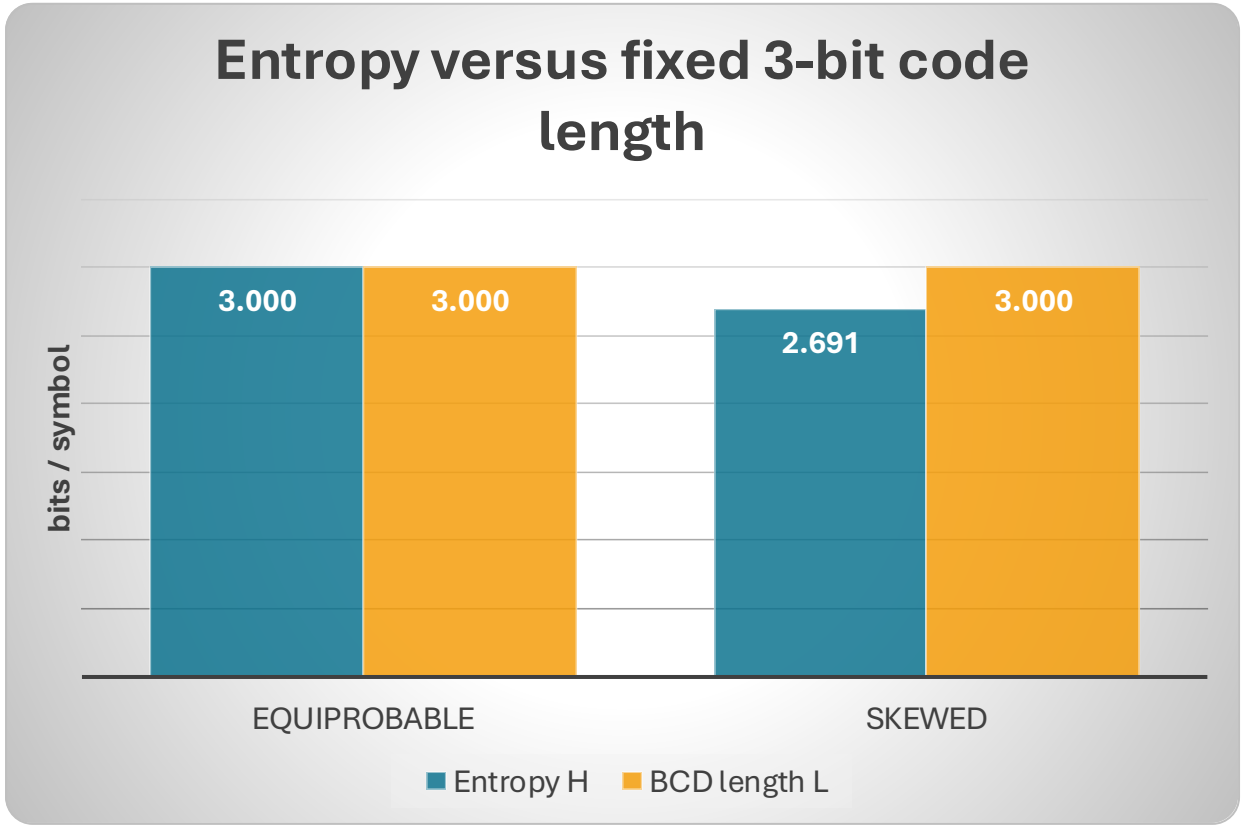
Shannon's source coding theorem

A code exists with L as close to H as we like: exactly H when every self-information is a whole number, and within one bit otherwise.

WORKED EXAMPLE 2.1

Same alphabet, two probability profiles

Eight symbols BCD-coded with fixed 3-bit codewords 000 … 111, so L = 3 bits/symbol in both cases.



Equiprobable

$p_i = 1/8$ for every i

$H = \log_2 8 = 3.0000$

$CE = 3 / 3 = 100\ \%$

Non-equiprobable

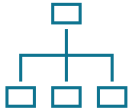
$p = (.27\ .20\ .17\ .16\ .06\ .06\ .04\ .04)$

$H = 2.6906$

$CE = 2.6906 / 3 = 89.69\ \%$

10.31 points lost to skew alone — exactly what Shannon-Fano and Huffman recover.

Split the sorted list from the top down



1 Sort

Order the symbols by descending probability.

2 Split

Divide the group into two halves of nearly equal total probability.

3 Label

Assign bit 0 to the top half and bit 1 to the bottom half.

4 Recurse

Repeat on each half until only one symbol remains.

FIRST SPLIT OF THE EIGHT-SYMBOL EXAMPLE

bit 0 · mass 0.47		bit 1 · mass 0.53					
m_1 .27	m_2 .20	m_3 .17	m_4 .16	m_5 .06	m_6 .06	m_7 .04	m_8 .04

Reading each symbol's bits from the top down gives its codeword — and the result is automatically a prefix code.

Eight symbols, four splits

Symbol	p_i	I_i	1	2	3	4	Codeword
m_1	.27	1.89	0	0			00
m_2	.20	2.32	0	1			01
m_3	.17	2.56	1	0	0		100
m_4	.16	2.64	1	0	1		101
m_5	.06	4.06	1	1	0	0	1100
m_6	.06	4.06	1	1	0	1	1101
m_7	.04	4.64	1	1	1	0	1110
bit 0 · top half			bit 1 · bottom half		already split off		1111

MASS BY CODEWORD LENGTH

0.47 — m_1, m_2
of the mass uses 2 bits

0.33 — m_3, m_4
of the mass uses 3 bits

0.20 — m_5 – m_8
of the mass uses 4 bits

2-bit code | 3-bit | 4-bit

$$L = 0.47(2) + 0.33(3) + 0.20(4) = 2.73 \text{ bits/symbol} \qquad CE = 2.6906 / 2.73 = 98.56\ \%$$

When probabilities are powers of two

Sym	p_i	I_i	1	2	3	4	5	6	7	bits
A	1/2	1	0							1
B	1/4	2	1	0						2
C	1/8	3	1	1	0					3
D	1/16	4	1	1	1	0				4
E	1/32	5	1	1	1	1	0			5
F	1/64	6	1	1	1	1	1	0		6
G	1/128	7	1	1	1	1	1	1	0	7
H	1/128	7	1	1	1	1	1	1	1	7

0 · symbol exits here | 1 · still in the group | already coded

A STRAIGHT COMB

Every split is exactly even, so at each bit position one symbol drops out with a 0 and the rest carry a 1 forward. No bit is ever wasted.

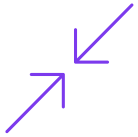
$$H = L = 1.984 \text{ bits/sym}$$

$$CE = 100\ \%$$

H is the last leaf, so it keeps all seven 1s

A fixed 3-bit code for the same eight symbols gives CE = 1.984 / 3 = 66 % — a third of the bandwidth wasted.

Merge the least probable items from the bottom up



1 Sort

Arrange symbols and subgroups by descending probability.

2 Merge

Combine the two least probable items into one new subgroup.

3 Label

Assign 0 to the higher-probability branch and 1 to the lower.

4 Repeat

Continue on the shortened list until a single item remains.

SHANNON-FANO

Top-down: repeatedly split the sorted list to balance probability mass.

HUFFMAN

Bottom-up: repeatedly merge the two least probable items.

READING THE CODEWORDS

Collect the 0s and 1s along the path from the root down to each leaf. The tree is built bottom-up, so the bit order must be reversed — skip that and the prefix property can break.

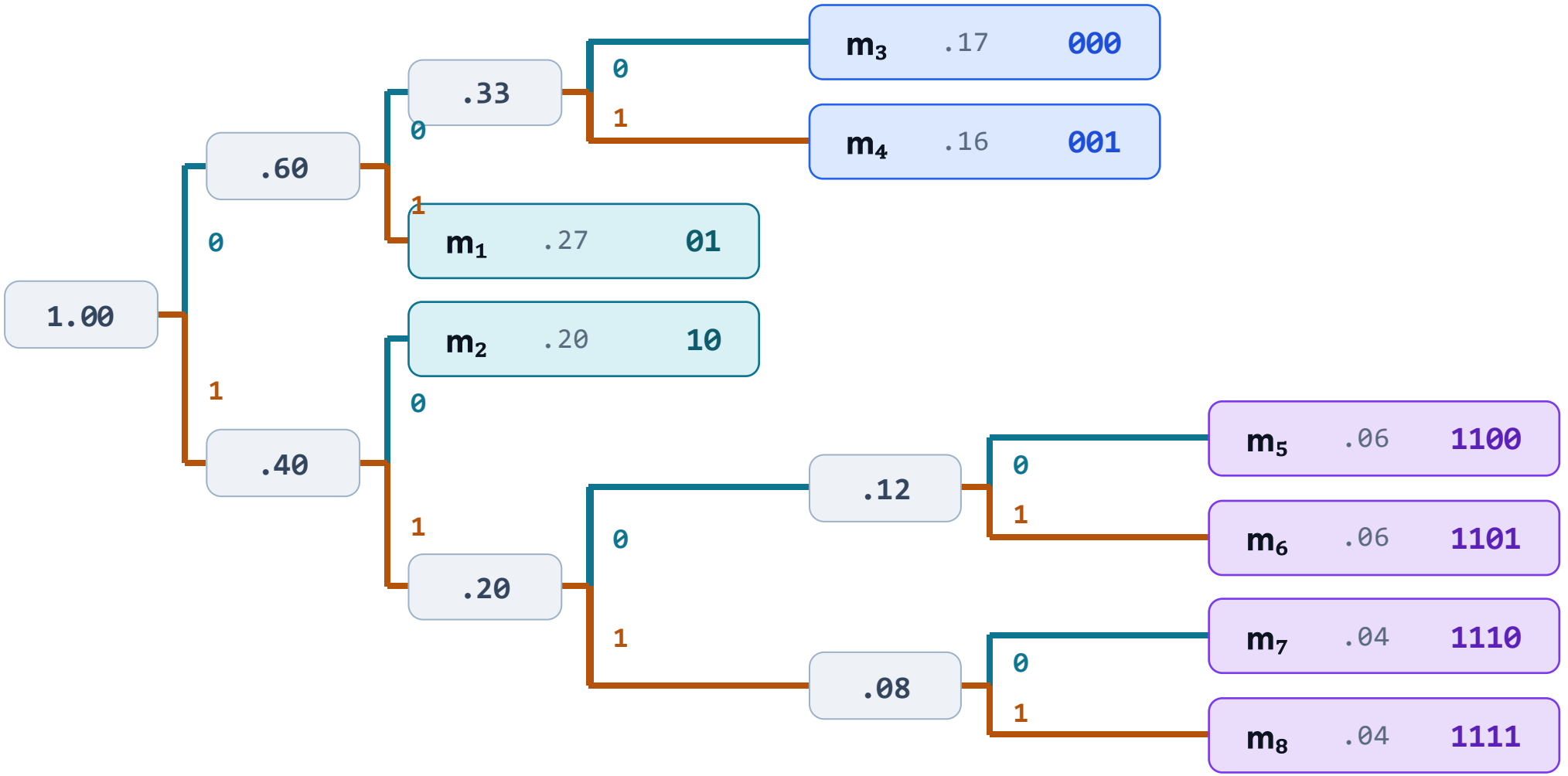
Seven merges for eight symbols

Start: m_1 .27 m_2 .20 m_3 .17 m_4 .16 m_5 .06 m_6 .06 m_7 .04 m_8 .04
higher probability → bit 0, lower → bit 1

1	m_7 (0) + m_8 (1)	→	m_{78} = 0.08	
2	m_5 (0) + m_6 (1)	→	m_{56} = 0.12	
3	m_{56} (0) + m_{78} (1)	→	m_{5678} = 0.20	
4	m_3 (0) + m_4 (1)	→	m_{34} = 0.33	
5	m_2 (0) + m_{5678} (1)	→	$m_{2,5678}$ = 0.40	tie at 0.20
6	m_{34} (0) + m_1 (1)	→	$m_{1,34}$ = 0.60	
7	$m_{1,34}$ (0) + $m_{2,5678}$ (1)	→	root = 1.00	tree complete

Root to leaf gives the codeword

bit 0 = higher-probability branch bit 1 = lower



Grey nodes are merged subgroups; purple leaves are symbols.

Reading the tree, then reversing the bits

Symbol	p_i	Codeword	bits
m_1	.27	01	2
m_2	.20	10	2
m_3	.17	000	3
m_4	.16	001	3
m_5	.06	1100	4
m_6	.06	1101	4
m_7	.04	1110	4
m_8	.04	1111	4

Example: m_5 sits at root(1) → (1) → (0) → (0), giving 1100.

SAME LENGTH DISTRIBUTION

Two 2-bit, two 3-bit and four 4-bit codewords — identical to Shannon-Fano on this alphabet.

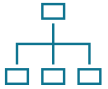
$$L = 0.47(2) + 0.33(3) + 0.20(4) = 2.73$$

$$CE = 2.6906 / 2.73 = 98.56\ \%$$

The two methods reach the same efficiency here, though in general which symbol gets which specific bit string can differ.

Greedy split versus provable optimum

Shannon-Fano



`top-down · split`

Repeatedly split the sorted list so each half carries roughly equal probability mass. Easy to reason about by hand, but the greedy split is not always globally optimal.

Usually close to optimal

Huffman

`bottom-up · merge`

Repeatedly merge the two least probable items. Provably produces the shortest possible average length among all prefix codes for the given probabilities.

Provably optimal

WHAT THEY SHARE

Both are entropy coding methods, and both produce prefix codes — so the encoded stream is always uniquely decodable.

Practical limit: for an alphabet of q symbols the longest codeword can reach q − 1 bits, which means large decoding buffers and high latency for big alphabets. In practice the two efficiencies are nearly identical.

SUMMARY

Key formulas from Lecture 2

Maximum entropy

$$H_{max} = \log_2 q$$

at $p_i = 1/q$, by Lagrange multipliers

Average codeword length

$$L = \sum p_i \ell_i$$

bits / symbol

Coding efficiency

$$CE = \eta = H / L$$

≤ 1, equals 1 when $L = H$

Design rule

$$\ell_i \approx -\log_2 p_i$$

match length to information content

Shannon-Fano

top-down split

balance probability mass at each step

Huffman

bottom-up merge

provably optimal among prefix codes

NEXT: LECTURE 3 — SOURCES WITH MEMORY

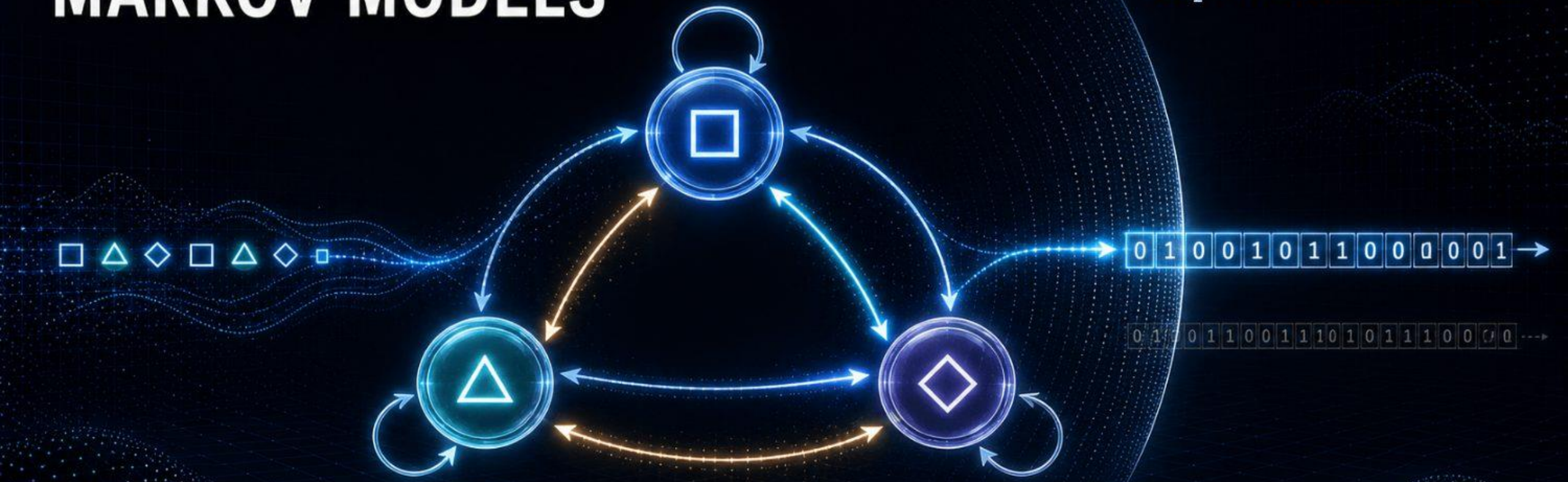
Markov models for correlated sources, where the dependence between successive symbols opens up gains beyond anything a memoryless entropy code can reach.

ELEC3203 · DIGITAL CODING & TRANSMISSION

LECTURE 3

SOURCES WITH MEMORY MARKOV MODELS

WINTEL
Wireless Intelligence & Networked Sensing Laboratory
Prof. Abdulkadir Çelik
https://akadircelik.com



Lecture Notes | Podcast | Table of Contents

Roadmap

Correlation · Markov models · entropy with memory · predictive coding

01 **Redundancy in real sources**

Speech, images and text are correlated, not independent

02 **Modelling memory**

First-order Markov chains, transition matrices, model order

03 **Entropy of a Markov source**

State entropy H_i, then averaged into $H = \sum P_i H_i$

04 **Worked example**

A two-state source, and how $H(k)$ converges to H

05 **Memoryless vs memory**

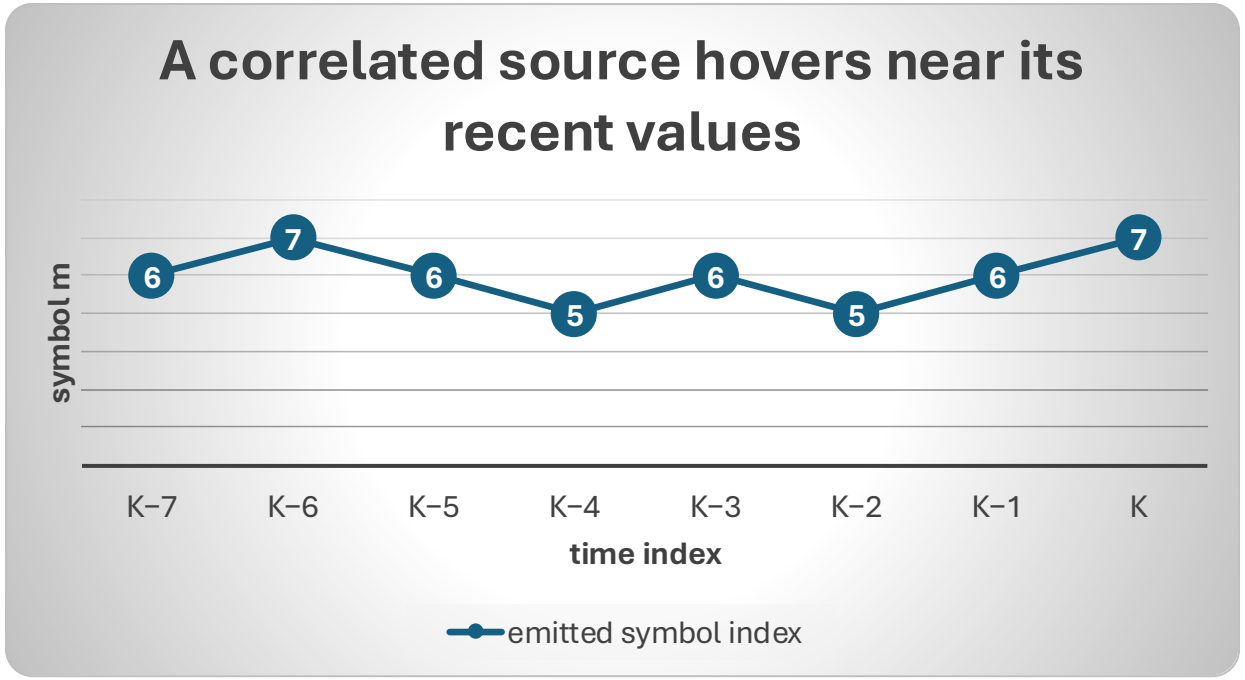
Why entropy coding a memory source directly wastes bits

NOTATION

New symbols in this lecture

X_i	the i-th state of the Markov source (equivalently symbol m_i)
P_i	prior (stationary) probability of being in state X_i
p_{ij}	transition probability $P(S(k)=j \mid S(k-1)=i)$
Γ	the transition probability matrix, with entries p_{ij}
H_i	entropy of the symbols emitted while in state X_i
H	overall source entropy, $H = \sum P_i H_i$
$H(k)$	average information per symbol over length-k blocks; $H(k) \to H$

Real signals are not memoryless



Speech

A sampled waveform changes slowly relative to the sampling rate, so consecutive samples sit close together.

Images and video

Neighbouring pixels look alike, and successive frames are almost identical.

Text

English has strong letter- and word-level correlation — a “q” is almost always followed by a “u”.

CORRELATION IS REDUNDANCY

Knowing the recent past narrows down what comes next, so a largely predictable symbol carries very little new information. Run Shannon-Fano or Huffman straight over a correlated sequence and all of that redundancy is thrown away.

Markov processes

The current symbol depends on a bounded window of the past, not the entire history.

First-order

$$P(S(k) \mid S(k-1))$$

Only the immediately preceding symbol matters. By far the most widely used model.

N-th order

$$P(S(k) \mid S(k-1), \ldots, S(k-N))$$

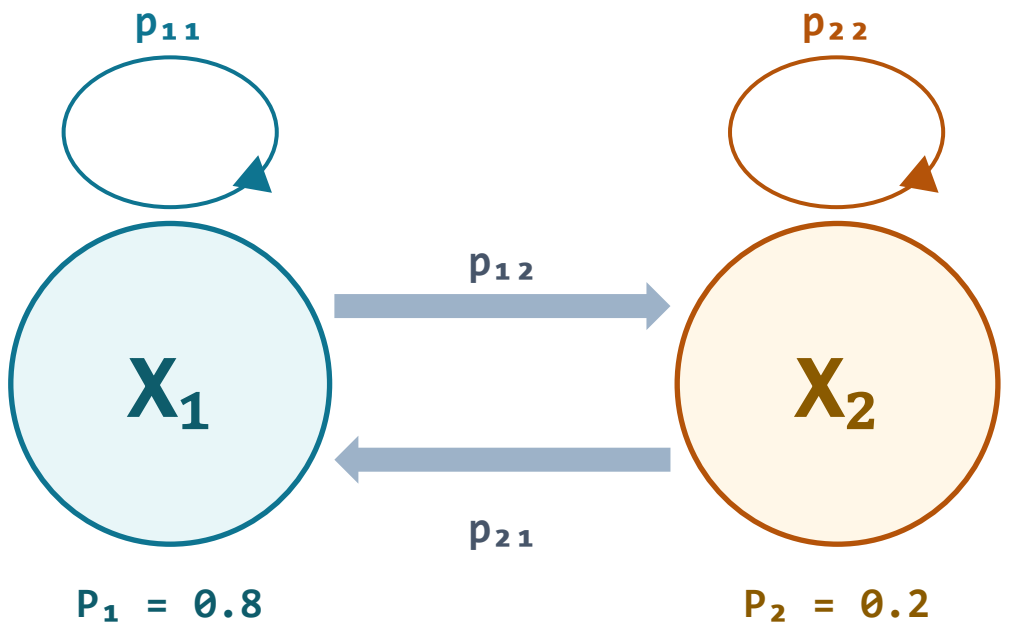
The N preceding symbols matter. More faithful, but the parameter count explodes.

PARAMETER COUNT FOR AN 8-SYMBOL ALPHABET (q^{N+1})

64	512	4 096	32 768
order 1 transition probabilities	order 2 transition probabilities	order 3 transition probabilities	order 4 transition probabilities

Longer memory is handled with a simplified parametric model: estimate only the conditional mean E[s(k) | past] and use it as a prediction, instead of the full conditional distribution.

States, transitions and the matrix Γ



Prior probabilities say how often the source sits in each state.

TRANSITION MATRIX Γ

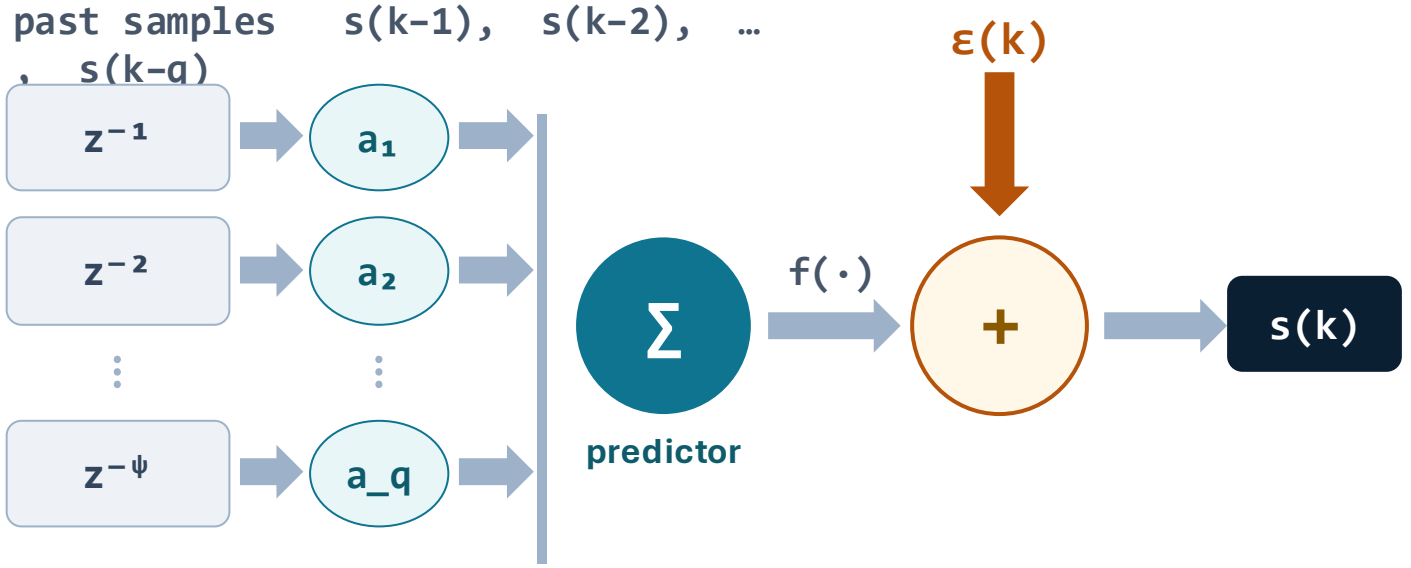
	to X_1	to X_2
from X_1	p_{11}	p_{12}
from X_2	p_{21}	p_{22}

$$p_{i1} + p_{i2} = 1 \quad \text{for each row } i$$

Each row of Γ is the distribution the source draws from while in that state.

The stickier the state — the closer p_{11} and p_{22} sit to 1 — the more correlated the source.

An equivalent view of memory



Weighted past samples form a prediction; adding the innovation reconstructs s(k) exactly.

THE MODEL

$$S(k) = f(S(k-1),\ldots) + \varepsilon(k)$$

LINEAR AR SPECIAL CASE

$$S(k) = \sum a_j\, S(k-j) + \varepsilon(k)$$

A good predictor makes the innovation ε(k) nearly uncorrelated and zero-mean, so almost all the redundancy has been absorbed into f(·).

Code the coefficients {a_j} and the innovation {ε(k)} instead of {s(k)} — the basis of predictive speech coding.

Per state first, then averaged

1 State entropy

the memoryless formula applied to row i of Γ

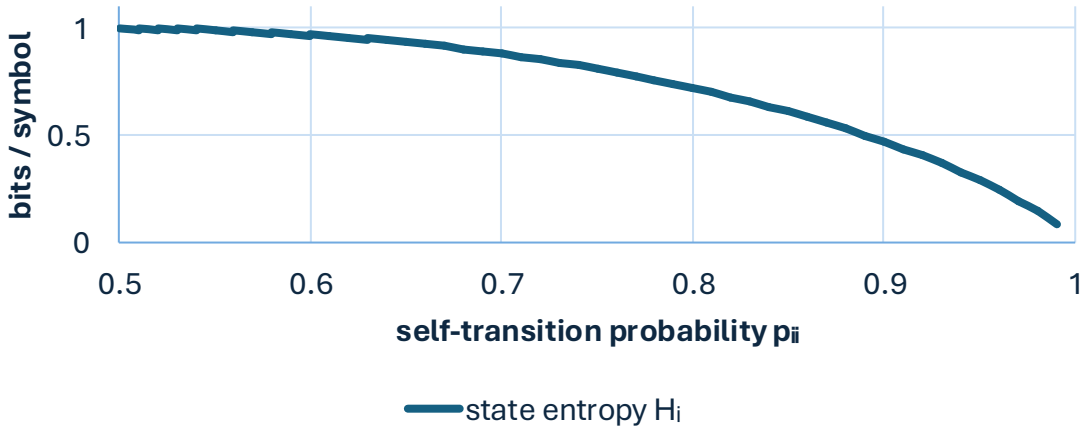
$$H_i = -\sum_j p_{ij} \log_2 p_{ij}$$

2 Overall entropy

state entropies weighted by how often each occurs

$$H = \sum_i P_i H_i$$

Stickier state, smaller state entropy



$$R = R_s \cdot H \quad \text{bits / second}$$

THE KEY QUALITATIVE FEATURE

The stickier the states, the more each row of Γ concentrates on a single outcome, so every H_i shrinks — and H with it. A highly correlated source carries less new information per symbol than an uncorrelated one built from the same alphabet.

3.4 · WORKED EXAMPLE 3.1

State entropies of a two-state source

State X_1

$P_1 = 0.8$ · row (0.9, 0.1)

$H_1 = -0.9 \log_2 0.9 - 0.1 \log_2 0.1 = 0.1368 + 0.3322$

$H_1 = 0.4690$ bits / symbol

State X_2

$P_2 = 0.2$ · row (0.4, 0.6)

$H_2 = -0.4 \log_2 0.4 - 0.6 \log_2 0.6 = 0.5288 + 0.4422$

$H_2 = 0.9710$ bits / symbol

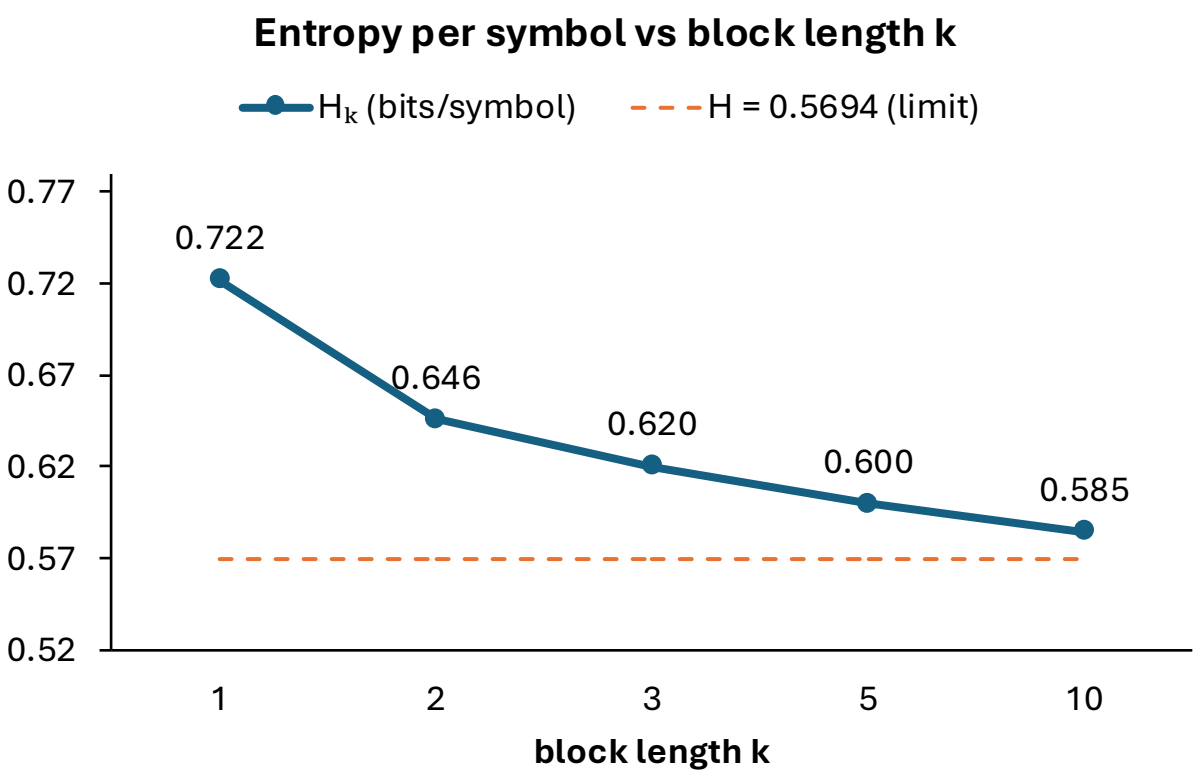
AVERAGE OVER THE STATES

$H = \sum_i P_i H_i = 0.8(0.4690) + 0.2(0.9710) = 0.5694$ bits/symbol — at $R_s = 1000$ symbols/s the information rate is $R = 569.4$ bits/s

Stationarity check: $P_1 = 0.8(0.9) + 0.2(0.4) = 0.8$ and $P_2 = 0.8(0.1) + 0.2(0.6) = 0.2$, so these priors are the ones the chain settles into.

Longer blocks reveal the memory

Group the output into blocks of k symbols and measure the entropy per symbol, H_k. Each extra symbol of context strips away more of the correlation a memoryless estimate ignores.



Memoryless estimate

priors only, k = 1

$H_1 = 0.7219$ bits / symbol

27 % above the true rate

True entropy rate

first-order chain, k → ∞

$H = 0.5694$ bits / symbol

the honest bit budget

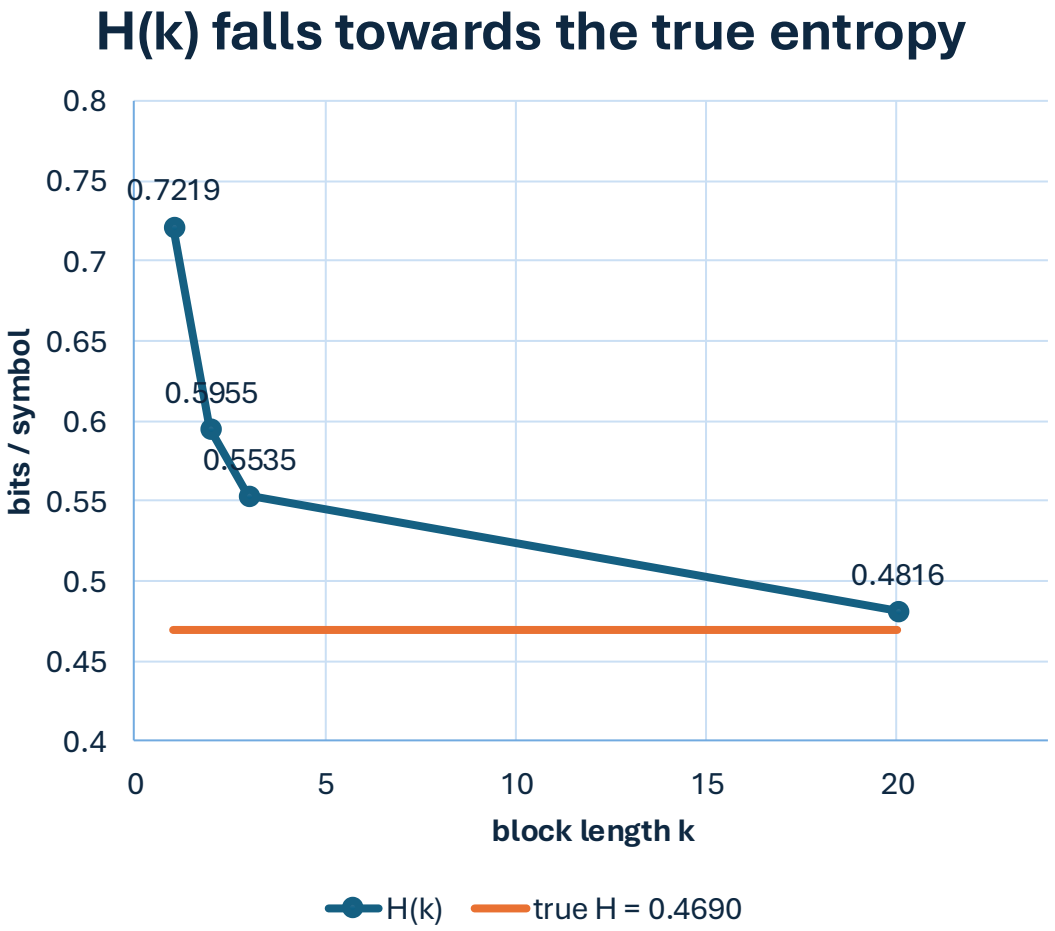
The curve flattens onto H, and the gap $H_1 - H = 0.1525$ bits/symbol is the redundancy a memoryless coder throws away.

Building H(3) from the probability tree

Sequence	Built from	Probability
111	$P(11) \cdot p_{11} = 0.72 \times 0.9$	0.648
112	$P(11) \cdot p_{12} = 0.72 \times 0.1$	0.072
121	$P(12) \cdot p_{21} = 0.08 \times 0.9$	0.072
122	$P(12) \cdot p_{22} = 0.08 \times 0.1$	0.008
211	$P(21) \cdot p_{11} = 0.18 \times 0.9$	0.162
212	$P(21) \cdot p_{12} = 0.18 \times 0.1$	0.018
221	$P(22) \cdot p_{21} = 0.02 \times 0.9$	0.018
222	$P(22) \cdot p_{22} = 0.02 \times 0.1$	0.002

Darker cells carry more of the probability mass.

$$-\sum p \log_2 p = 1.6606 \text{ bits per triple}$$
$$H(3) = 1.6606 / 3 = 0.5535$$



Why not just entropy-code it directly?

Same alphabet, same marginal probabilities, same symbol rate — but one source has memory.

Memoryless model

what a symbol-by-symbol coder can see

Huffman and Shannon-Fano only ever see the one-symbol statistics p_i, so the best they can reach is the equivalent memoryless entropy H(1).

$$H(1) = 0.7219 \text{ bits/sym}$$

Source with memory

the true physical property of the source

Correlation only ever reduces uncertainty about what comes next, never increases it — so the true entropy sits well below the memoryless estimate.

$$H = 0.4690 \text{ bits/sym}$$

$$H(ml) \gg H(m) \quad \Rightarrow \quad R(ml) \gg R(m)$$

more than 50 % too high, here

THE FIX

Predict S(k) from its past, subtract, and entropy-code only the near-memoryless innovation ε(k).

SUMMARY

Key formulas from Lecture 3

State entropy

$$H_i = -\sum_j p_{ij} \log_2 p_{ij}$$

row i of Γ, in bits/symbol

Overall entropy

$$H = \sum_i P_i H_i$$

weighted by the priors, and ≤ $\log_2 q$

Information rate

$$R = R_s \cdot H$$

bits / second

Block entropy

$$H(k) \downarrow H$$

decreases monotonically as k grows

Memory always helps

$$H(m_1) \geq H(m)$$

correlation never raises uncertainty

Model cost

$$q^{N+1} \text{ parameters}$$

why first-order chains dominate

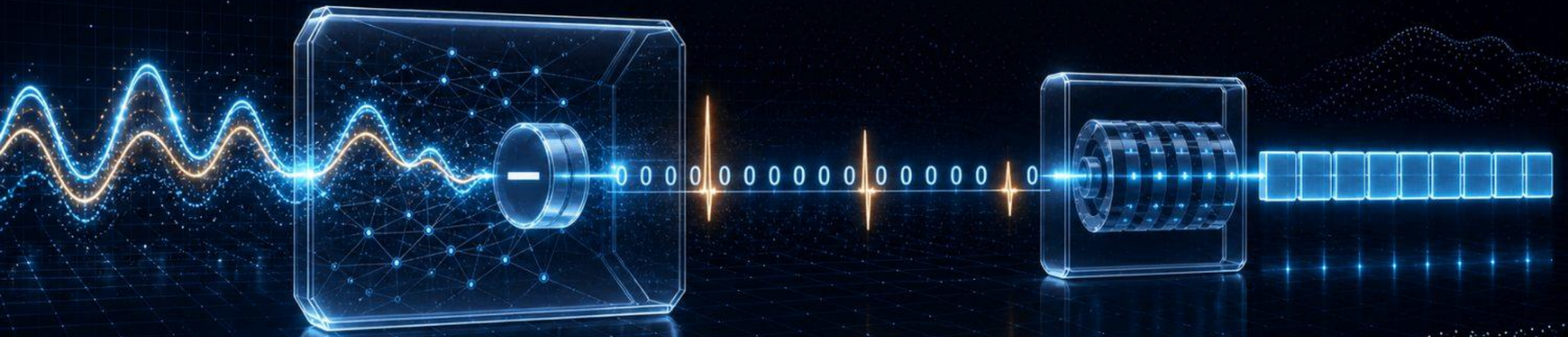
NEXT: LECTURE 4 — PREDICTIVE AND RUN-LENGTH CODING

Build a predictor, subtract it off, and compress the mostly-zero residual with run-length coding — turning this lecture's insight into a concrete coding scheme.

LECTURE 4

PREDICTIVE CODING AND RUN-LENGTH CODING

WINTEL
Wireless Intelligence & Networked Sensing Laboratory
Prof. Abdulkadir Çelik
https://akadircelik.com



Lecture Notes
Podcast
Table of Content

ELEC3203 · LECTURE 4

Roadmap

Redundancy removal · run-length coding · compression ratio

01 **Removing redundancy**

Why prediction must come before entropy coding

02 **A predictive scheme with RLC**

The full encoder / decoder chain with a shared predictor

03 **The RLC codeword table**

Runs of zeros indexed into fixed n-bit codewords

04 **Average length and ratio**

$d = (1 - p^N)/(1 - p)$ and $C = d / n$

05 **RLC vs Shannon-Fano vs Huffman**

Variable in / fixed out, and when each one wins

NOTATION

New symbols in this lecture

`x(i)`	binary source sequence, after converting the q-ary source to bits
`x^(i)`	the predictor's estimate of x(i) from its recent past
`e(i)`	residual, e(i) = x(i) − x̂(i) — mostly zeros if the prediction is good
`p`	P(“0”), the probability a residual bit is zero (assumed close to 1)
`l`	length of a run of zeros before a “1”, or before the end of a block
`n`	fixed length, in bits, of each RLC output codeword
`N`	$N = 2^n - 1$, the largest run length the codeword table can index
`d`	average input pattern length before coding, in bits
`C`	compression ratio, $C = d / n$

Predict first, then entropy-code

THE PROBLEM CARRIED OVER FROM LECTURE 3

Entropy-coding a memory source symbol by symbol only ever sees the one-symbol statistics, so it reaches $R_s H(1)$ — not the true, much smaller $R_s H$.

Attack the redundancy directly: predict x(i) from its past, subtract, and code only what is left.

Speech coding

Consecutive samples are highly correlated. An AR model removes the predictable part, leaving an innovation ε(k) that is close to uncorrelated and zero-mean.

send $\{a_j\}$ and ε(k), not s(k)

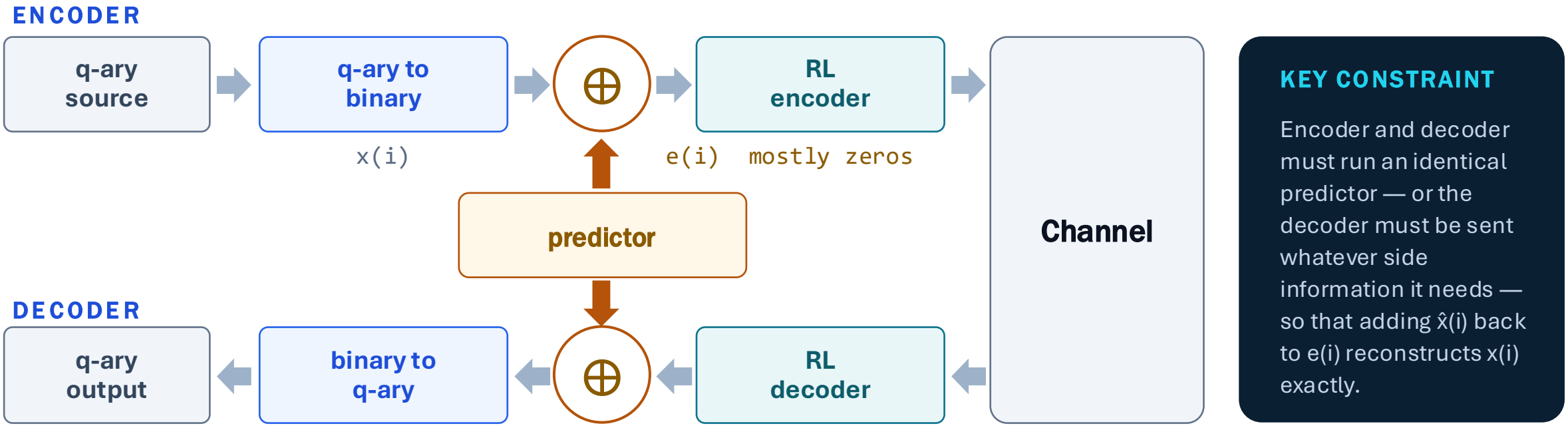
Video coding

A frame sequence carries both inter-frame redundancy in time and intra-frame redundancy in space. Removing both leaves a far smaller residual to code.

inter-frame + intra-frame

Once the predictable part is gone, the residual is binary and overwhelmingly zeros — exactly what run-length coding is built for.

The complete coding chain



KEY CONSTRAINT

Encoder and decoder must run an identical predictor — or the decoder must be sent whatever side information it needs — so that adding $\hat{x}(i)$ back to e(i) reconstructs x(i) exactly.

The residual is what travels: RLC compresses it before the channel, and the decoder adds the prediction back.

WHY RLC AND NOT HUFFMAN HERE

After a good predictor the residual is binary and overwhelmingly zero — a heavily skewed source that run-length coding handles with a single fixed lookup table.

Runs of zeros, indexed into fixed codewords

run length l	encoder input	output codeword	input bits
0	1	00…000	1
1	01	00…001	2
2	001	00…010	3
3	0001	00…011	4
⋮	⋮	⋮	⋮
N−2	0…01	11…101	N−1
N−1	00…01	11…110	N
max	00…00 (no 1)	11…111	N

Darker input cells = longer, rarer runs. Every output codeword is the same width.

Indexed by run length

$0 \le l \le N - 1,\quad N = 2^n - 1$

Input length: variable

$\min\{ N, l + 1 \}$ bits

Output length: fixed

exactly n bits

It all rests on one assumption: the bit stream is mostly zeros, so p = P(“0”) is close to 1 and short runs dominate.

Average input length and compression ratio

Each codeword is exactly n bits, so the only thing that varies is how many input bits it replaces.

AVERAGE OVER THE GEOMETRIC DISTRIBUTION OF RUN LENGTHS

$$d = \sum_{l=0}^{N-1} (l+1)\, p^l (1-p) + N\, p^N \qquad \text{for } l = 0 \ldots N-1$$

$$= 1 + p + p^2 + \cdots + p^{N-1} \qquad \text{a geometric series}$$

$$= (1 - p^N) / (1 - p) \qquad \text{closed form}$$

AVERAGE INPUT LENGTH

$$d = (1 - p^N) / (1 - p)$$

COMPRESSION RATIO

$$C = d / n = (1 - p^N) / n(1 - p)$$

CHOOSING n

A larger n lets the table index longer runs, raising N exponentially — but a single “1” still costs a full n-bit codeword, so n must be matched to how skewed p really is.

4.4 · WORKED EXAMPLE

p = 0.95 with 5-bit codewords

Setup

$p = 0.95, \quad n = 5$

$$N = 2^5 - 1 = 31$$

Average input length

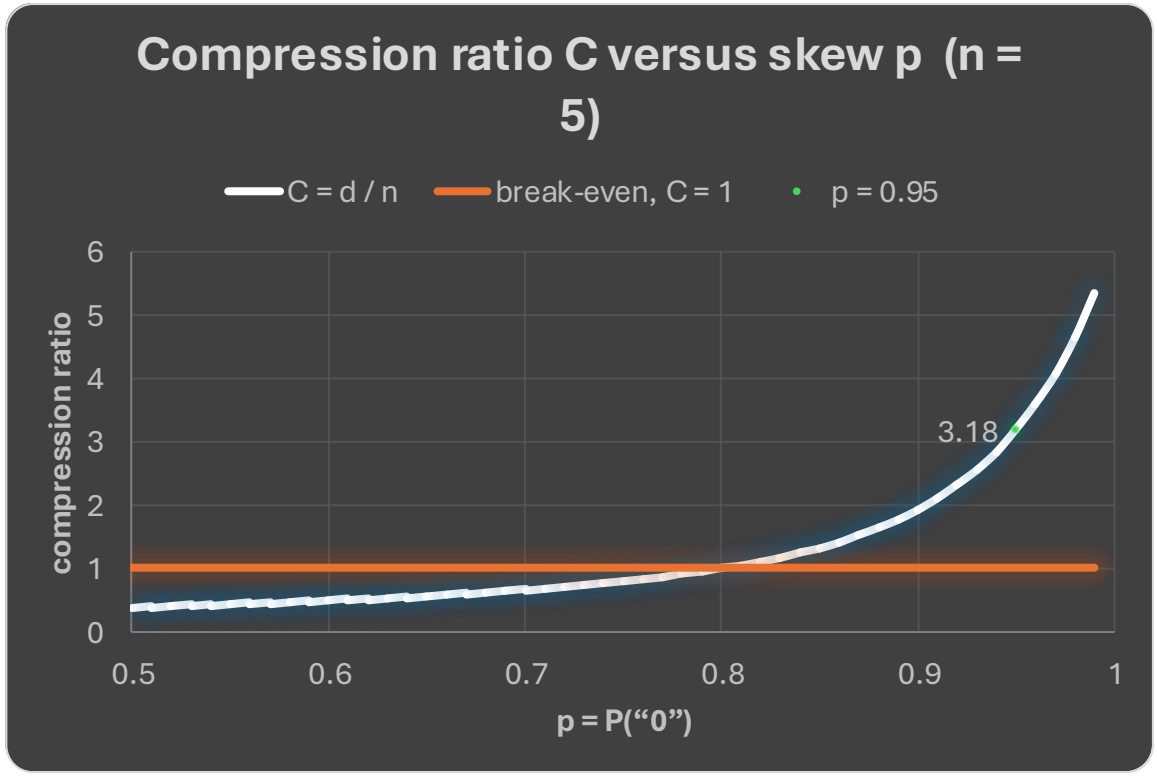
$d = (1 - 0.95^{31}) / 0.05$

$$= (1 - 0.2038) / 0.05 \approx 15.92 \text{ bits}$$

Compression ratio

$C = d / n = 15.92 / 5$

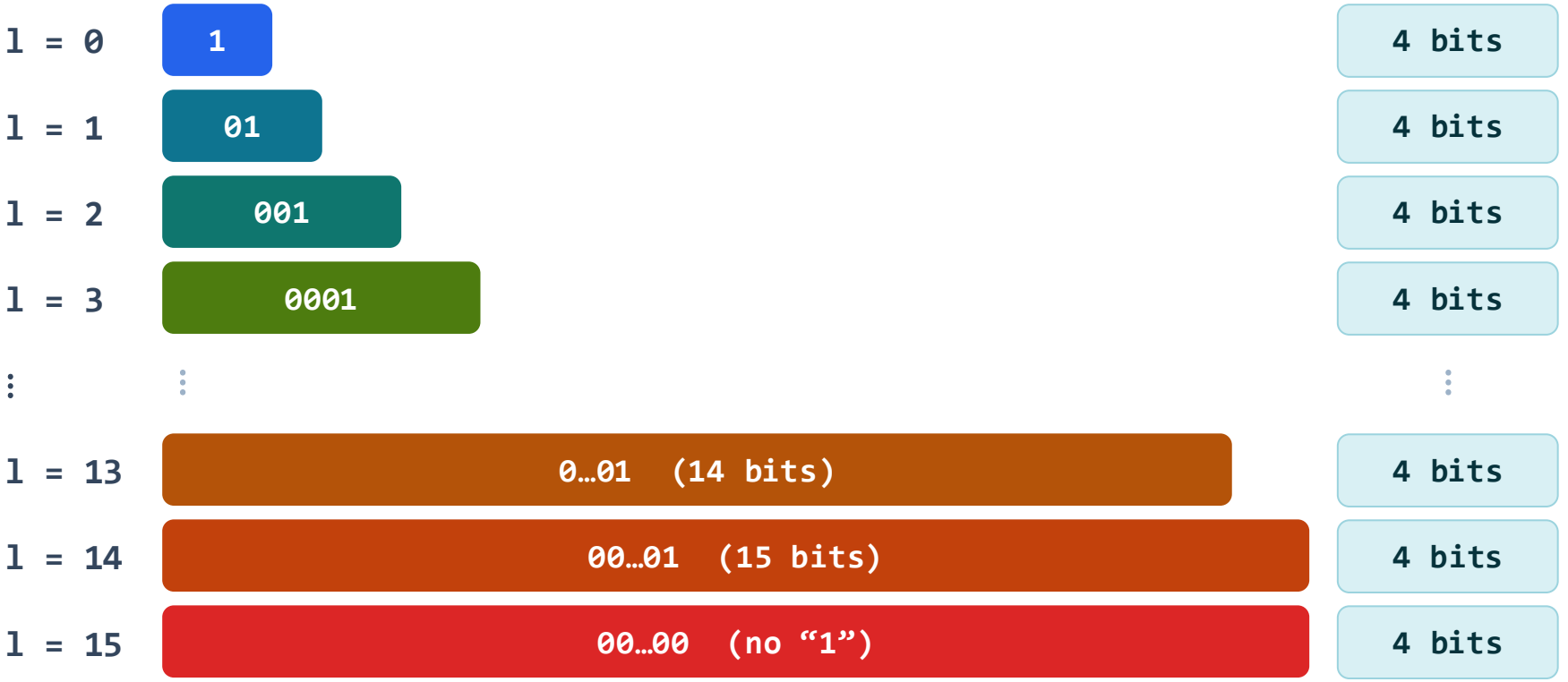
$$\approx 3.18\times$$



Roughly 15.9 bits of residual collapse into a single 5-bit codeword — a 3.18× reduction, for bits that are zero 95 % of the time.

The visual reverse of a Huffman tree

Illustrative case n = 4, so N = 15. Bar width is the input pattern length; colour shows how rare that run is.



SAME PRINCIPLE

RLC is variable in, fixed out. Shannon-Fano and Huffman are fixed in, variable out. Both obey one rule: a rare event maps to a relatively larger codeword.

Every input, however long, collapses to the same 4-bit output — and all three schemes are lossless entropy encodings.

Which side is held to a fixed length?

	Run-length coding	Shannon-Fano	Huffman
Input length	variable, 1 to N bits	fixed, one symbol	fixed, one symbol
Output length	fixed, n bits	variable, $\approx -\log_2 p_i$	variable, $\approx -\log_2 p_i$
Source targeted	binary, heavily skewed ($p \to 1$)	general q-ary, any distribution	general q-ary, any distribution
Construction	fixed lookup table, indexed by run length	top-down recursive splitting	bottom-up greedy merging
Optimality	not optimal in general, but very cheap	usually close to optimal	provably optimal among prefix codes
Typical use	residual after prediction; fax; simple images	quick hand-built codes; teaching	general-purpose, inside JPEG and ZIP

All three are lossless entropy codings; they differ mainly in which side of the mapping is held to a fixed length.

SUMMARY

Key formulas from Lecture 4

Table size

$N = 2^n - 1$

largest run length the table can index

Input pattern length

$\min\{ N, l + 1 \}$

bits — variable

Average input length

$d = (1 - p^N)/(1 - p)$

bits, over the run-length distribution

Compression ratio

$C = d / n$

output is always n bits

The assumption

$p = P(\text{“0”}) \to 1$

short runs must dominate

Order of operations

predict, then code

never entropy-code a memory source raw

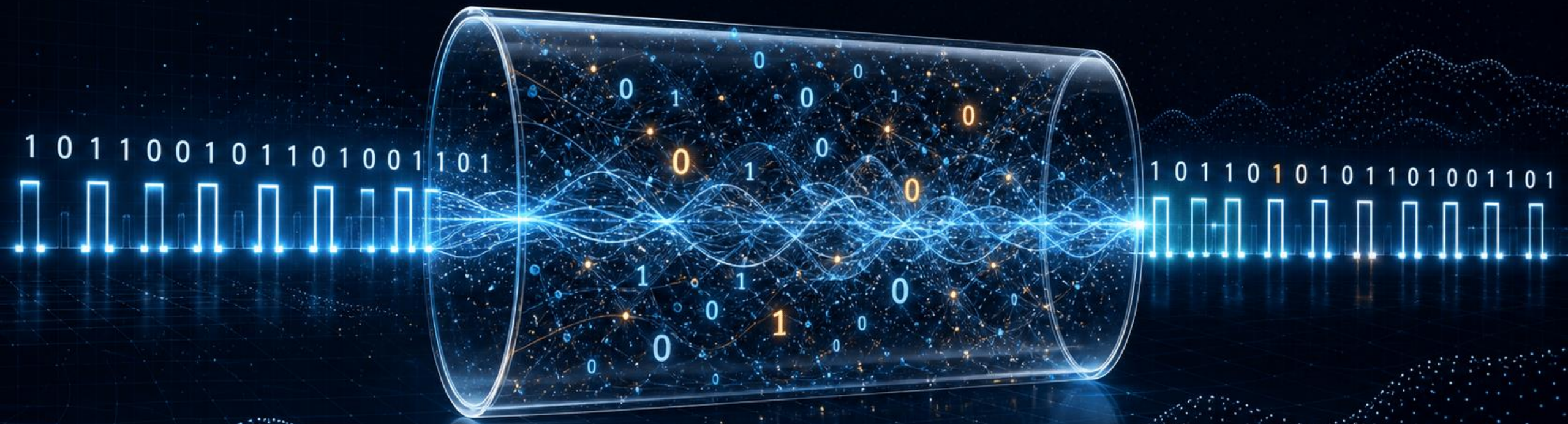
NEXT: LECTURE 5 — CHANNELS AND MUTUAL INFORMATION

The source side is done. Next comes the channel: the AWGN model, the binary symmetric channel, and what a receiver actually learns once noise is in the picture.

LECTURE 5

CHANNELS AND MUTUAL INFORMATION

WINTEL
Wireless Intelligence & Networked Sensing Laboratory
Prof. Abdulkadir Çelik
https://akadircelik.com



Lecture Notes
Podcast
Table of Content

ELEC3203 · LECTURE 5

Roadmap

Channel models · AWGN · binary symmetric channel · mutual information

01 **Modelling the channel**

The ideal channel, AWGN, real propagation — then BPSK and detection turn it into a BSC

02 **The binary symmetric channel**

What AWGN becomes after a hard decision: one crossover probability p_e

03 **Mutual information of a pair**

What observing Y_j tells you about X_i — and when it misleads

04 **Average mutual information**

$I(X,Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$

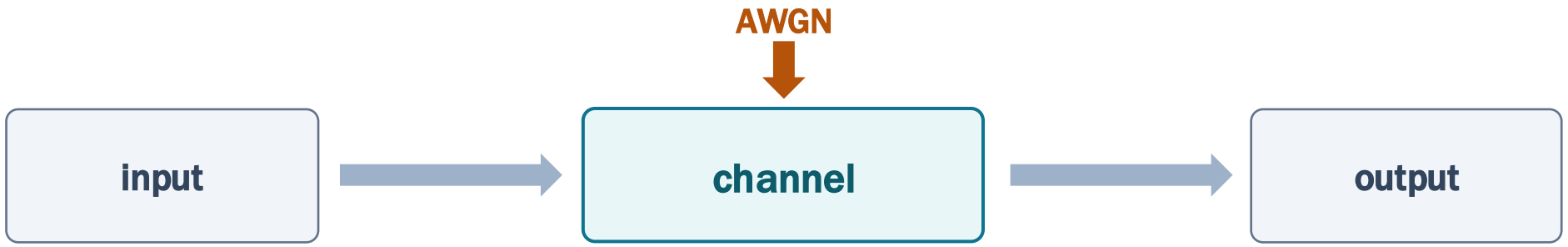
Lectures 1 to 4 asked how efficiently a source can be coded. This one asks how much of it survives the channel.

NOTATION

New symbols in this lecture

X_i	a channel input symbol — what is transmitted
Y_j	a channel output symbol — what is received
$P(Y_j \mid X_i)$	probability of receiving Y_j given X_i was sent
p_e	the crossover (error) probability of a binary symmetric channel
$I(X_i, Y_j)$	mutual information of one input/output pair, in bits
$I(X, Y)$	average mutual information, in bits/symbol
$H(X)$	source entropy — average information sent
$H(X \mid Y)$	equivocation — average information about X lost in the channel
$H(Y),\ H(Y \mid X)$	destination entropy, and error entropy injected by noise

An ideal path, corrupted only by noise



Real channels distort amplitude and phase and have finite bandwidth B — here we keep the ideal path plus noise.

THE IDEAL CHANNEL

$h(t) = \delta(t - T)$ a pure delay

$A(\omega) = 1$ flat amplitude to B

$\Phi(\omega) = -\omega T$ linear phase

ADDITIVE WHITE GAUSSIAN NOISE

zero-mean Gaussian, variance σ^2

flat PSD $N_0/2$ across bandwidth B

noise power $N_p = N_0 B$

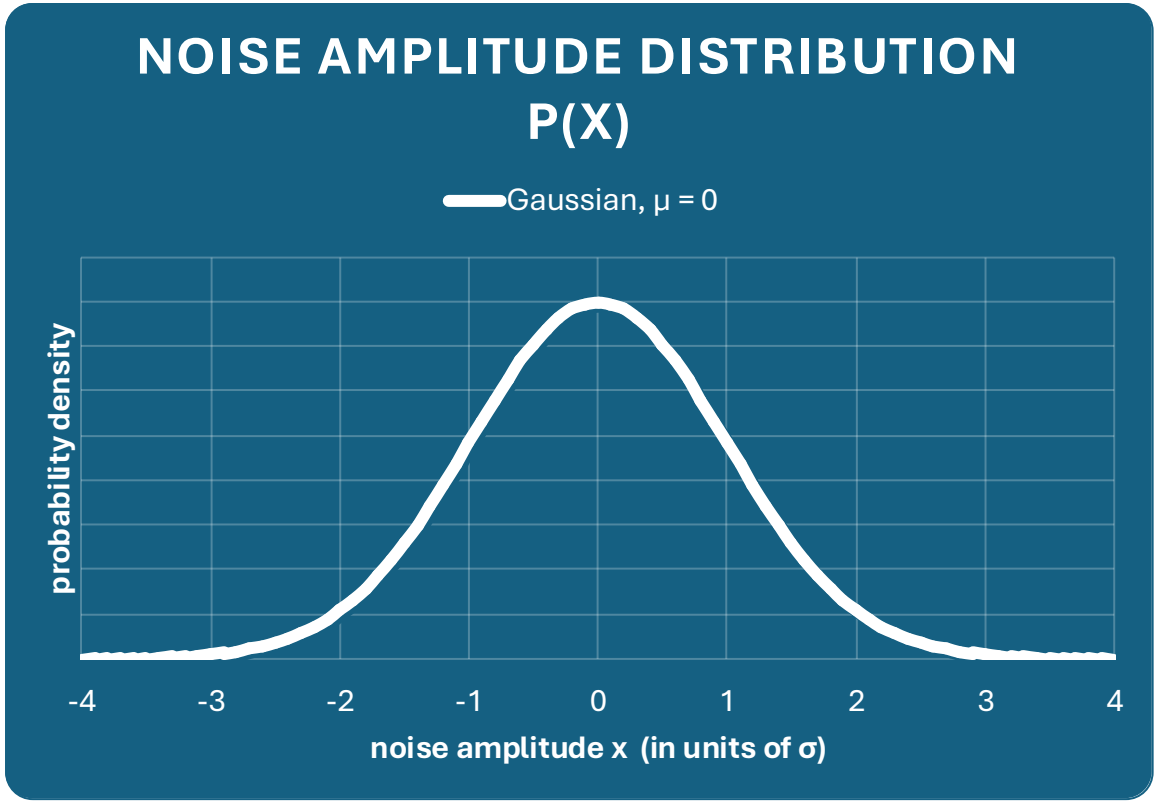
WHERE THE NOISE COMES FROM

The received signal is weak and must be amplified before detection — and the amplifier injects thermal noise, set by its noise figure.

MEMORYLESS OR DISPERSIVE

A dispersive channel has memory and adds intersymbol interference. We work with the memoryless, non-dispersive case.

Gaussian in amplitude, white in frequency



Gaussian in amplitude

$p(x) = (1/\sqrt{(2\pi)}\sigma)\ e^{-(x-\mu)^2/2\sigma^2}$

zero-mean, μ = 0, with σ^2 equal to the noise power

White in frequency

$N(\omega) = N_0/2 \quad \text{for all } \omega$

equal noise power at every frequency, so $N_p = N_0B$

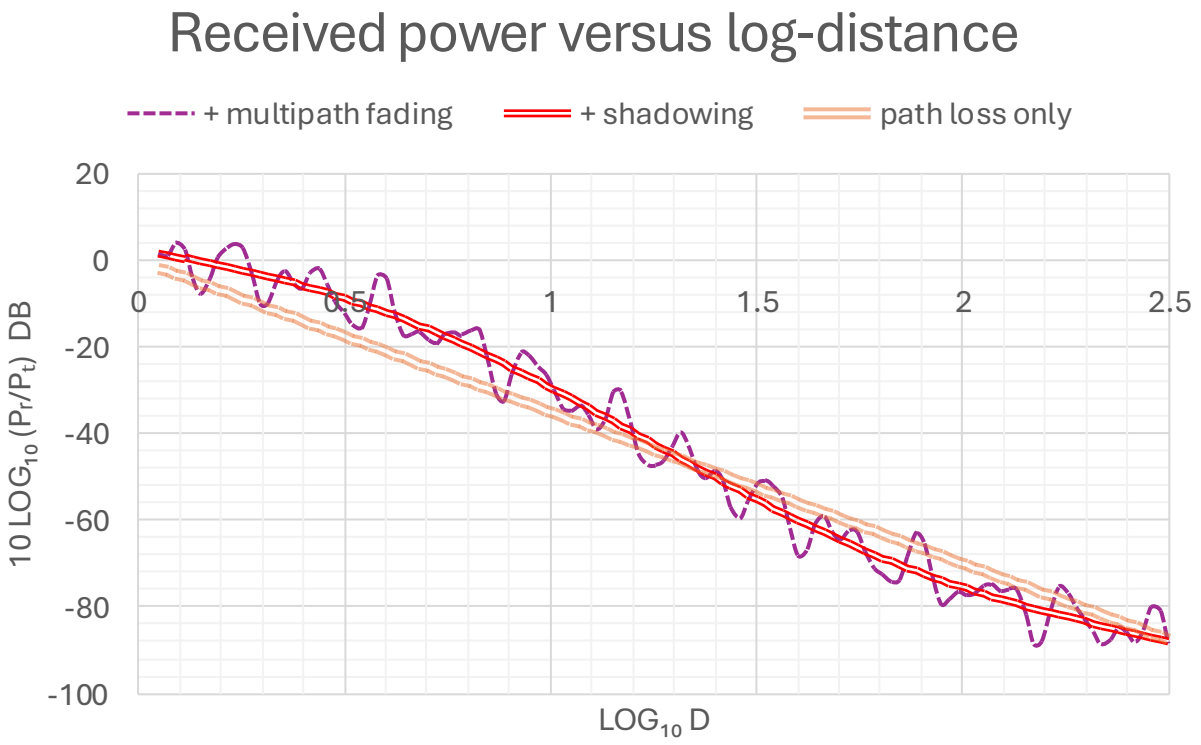
Impulse autocorrelation

$R(\tau) = 0 \quad \text{for } \tau \neq 0$

noise samples are uncorrelated from one instant to the next

PSD and autocorrelation are a Fourier pair: a flat spectrum in frequency is a spike in time — which is exactly why white noise samples are uncorrelated.

Three effects, three physical scales



Path loss

`large-scale · deterministic`

Average power falls smoothly with distance, $P_r/P_t \propto d^{-n}$ — linear against log d.

Shadowing

`medium-scale · random`

Buildings, terrain and foliage attenuate by different amounts, log-normally distributed.

Multipath fading

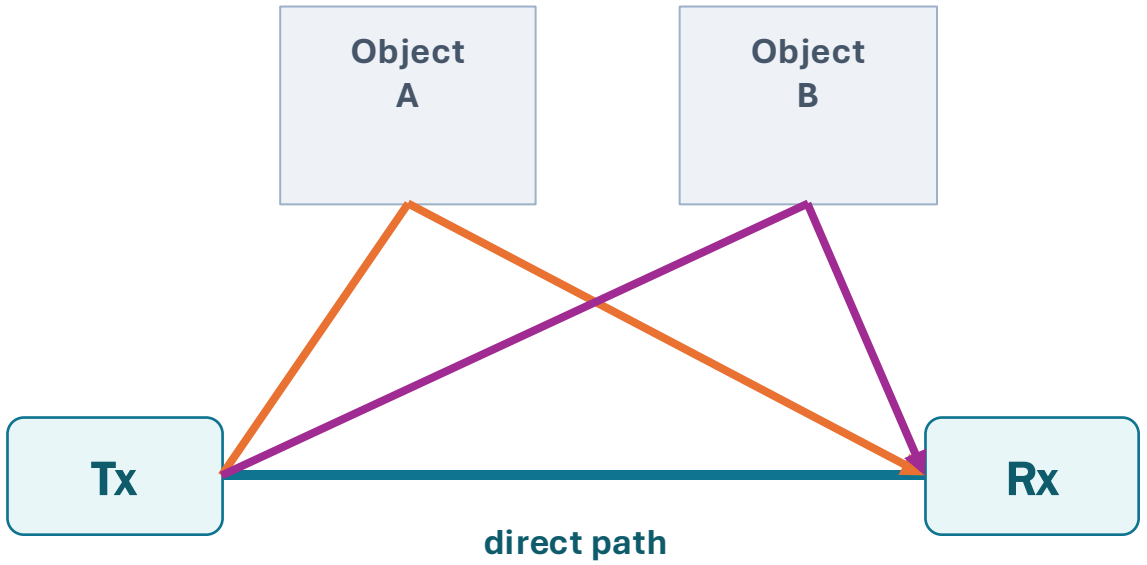
`small-scale · rapid`

Reflected copies arrive with different delays and phases and interfere, over a fraction of a wavelength.

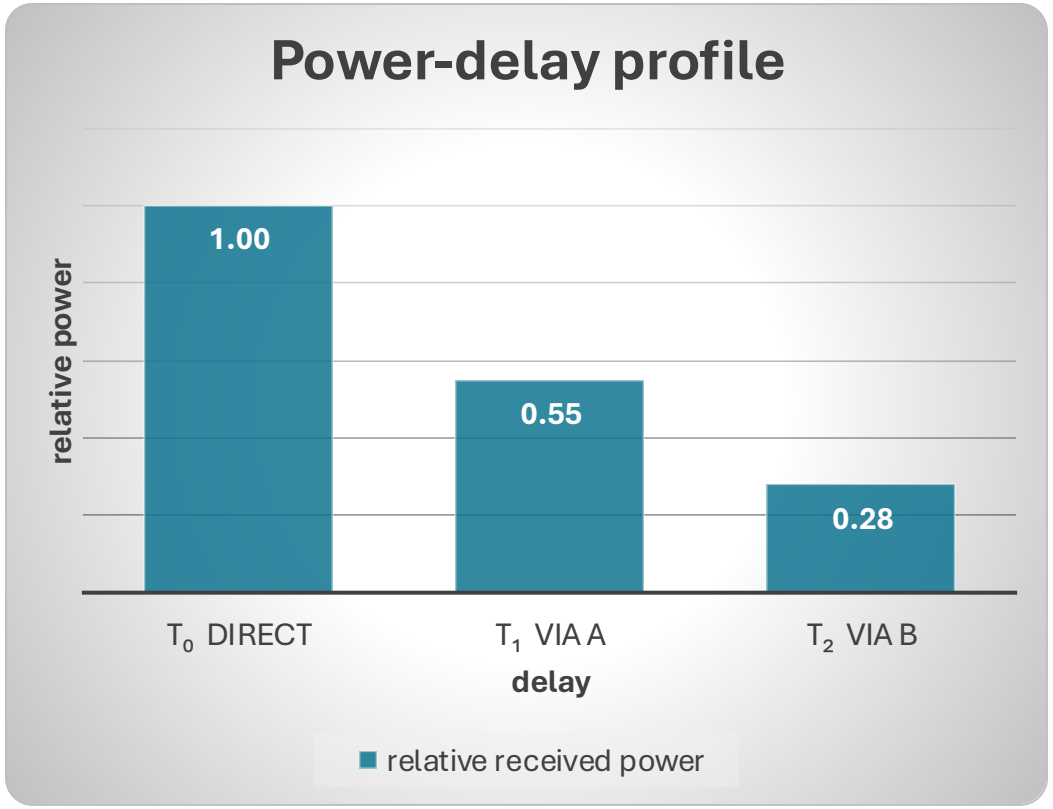
All three are superimposed on a real link. Only path loss is present in the idealised AWGN channel used for the rest of this lecture.

Many copies, many delays

SIGNAL PATHS



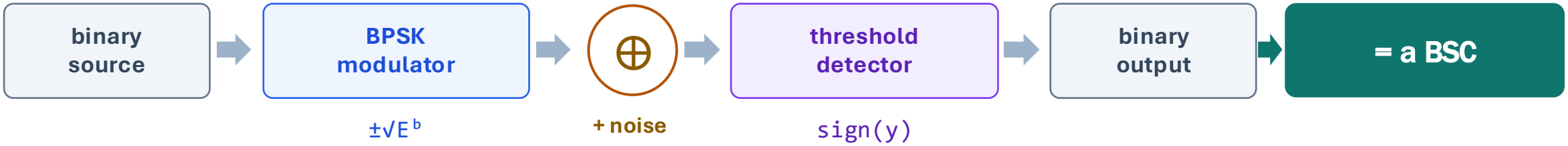
Reflections off A and B arrive later and weaker than the direct path.



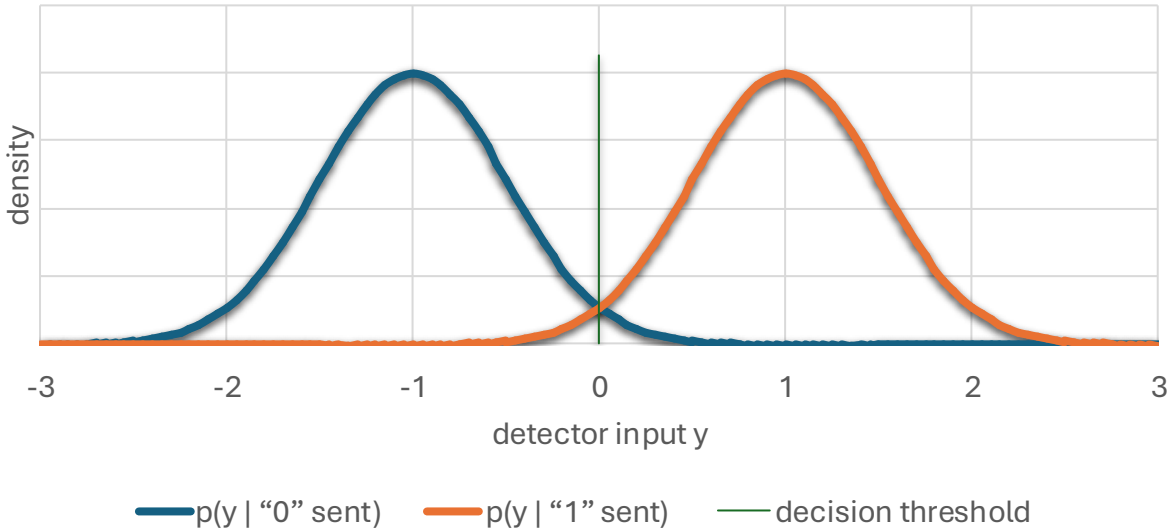
The receiver sums all of these copies — constructive and destructive interference is what produces rapid multipath fading.

Where the crossover probability comes from

The BSC is not a separate model — it is what the AWGN channel becomes once you add a modulator and a hard decision.



Overlapping tails are the bit errors



The hard decision

Decide “1” if y > 0, “0” otherwise. The continuous output y collapses to one bit.

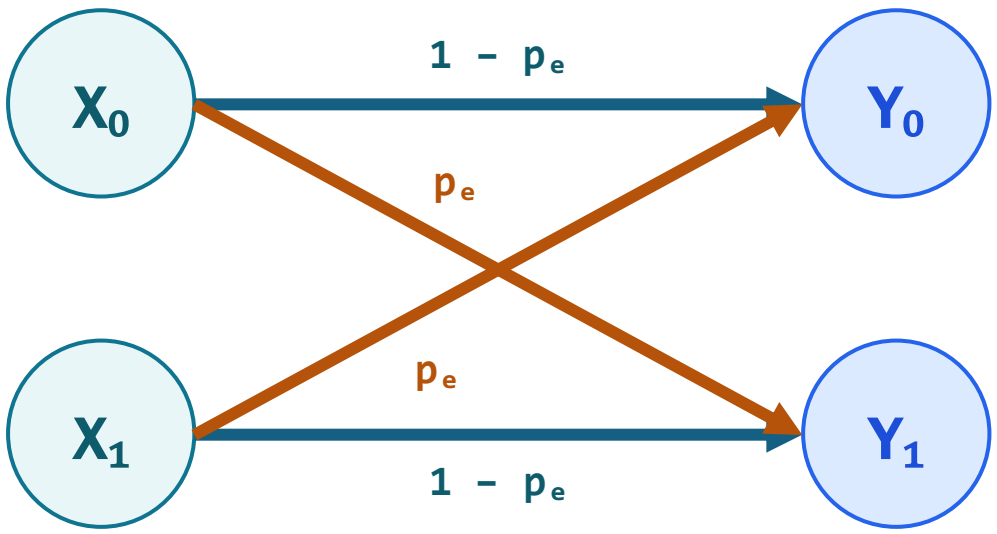
The crossover probability

$$p_e = Q(\sqrt{(2E^b/N_o)})$$

Why it is symmetric

The two conditional densities are mirror images about the threshold, so both error events have equal probability.

The discrete channel that AWGN becomes



Teal = correct reception, amber = the two symmetric error events.

SYMMETRIC MEANS

$$P(Y_1|X_0) = P(Y_0|X_1) = p_e$$

Both ways of getting it wrong are equally likely, so one number p_e describes the whole channel.

JOINT PROBABILITIES, BY BAYES

$$P(X_i, Y_j) = P(X_i)\,P(Y_j|X_i) = P(Y_j)\,P(X_i|Y_j)$$

The same joint event, read from either end of the channel.

Exactly the model the previous slide produced: AWGN + BPSK + a hard decision, with $p_e = Q(\sqrt{(2E^b/N_0)})$.

5.2 · WORKED EXAMPLE

A skewed source on a 2 % BSC

$P(X_0) = 0.3, \quad P(X_1) = 0.7$ (70 % of transmitted bits are "1"), $\quad p_e = 0.02$

THE FOUR JOINT PROBABILITIES

	received Y_0	received Y_1	$P(X_i)$
sent X_0	0.294	0.006	0.300
sent X_1	0.014	0.686	0.700
$P(Y_j)$	0.308	0.692	1.000

teal = correct reception

amber = error

Correct reception

$0.294 + 0.686 = 0.980 = 1 - p_e$

Erroneous reception

$0.006 + 0.014 = 0.020 = p_e$

Output probabilities

$P(Y_0) = 0.308, \quad P(Y_1) = 0.692$

READING THE TABLE

Grouped by outcome, the joints reproduce the channel's error rate exactly. Summed over the input instead, they give the output probabilities — which are slightly more balanced than the source.

What does observing Y_j tell you about X_i?

$$I(X_i, Y_j) = \log_2 [\, P(X_i \mid Y_j) / P(X_i) \,] \quad \text{bits}$$

Perfect channel

$Y_i = X_i, \quad p_e = 0$

$P(X_i \mid Y_i) = 1$

$I = \log_2 (1 / P(X_i)) = I(X_i)$

the full information content gets through

Useless channel

$p_e = 0.5$

Y_i independent of $X_i \Rightarrow P(X_i|Y_i) = P(X_i)$

$I = \log_2 1 = 0$

nothing gets through at all

SOURCE INFORMATION − CONVEYED = LOST

$I(X_i)$ − $I(X_i, Y_j)$ = $I(X_i \mid Y_j)$

information sent | conveyed to the receiver | destroyed in the channel

with $0 \le I(X_i \mid Y_j) \le I(X_i)$ — zero loss for a perfect channel, total loss when $p_e = 0.5$

Four pairs, two of them misleading

Bayes gives the reverse conditionals; the source contents are $I(X_0) = 1.737$ and $I(X_1) = 0.515$ bits.

pair	$P(X_i \mid Y_j)$	$I(X_i, Y_j)$ bits	joint prob.
$X_0 \to Y_0$	0.9545	**+1.670**	0.294
$X_1 \to Y_1$	0.9913	**+0.502**	0.686
$X_0 \to Y_1$	0.0087	**−5.113**	0.006
$X_1 \to Y_0$	0.0455	**−3.945**	0.014

correct pairs · near the source content

error pairs · negative, mis-information

AT THE DESTINATION

$I(Y_0) = 1.699$ bits

$I(Y_1) = 0.531$ bits

More balanced than the source (1.737 / 0.515): noise has evened out the symbol probabilities.

WHY THE NEGATIVE VALUES MATTER

The correct pairs deliver almost all of the source content — 1.670 of 1.737 bits, and 0.502 of 0.515. The error pairs are strongly negative: observing Y_1 when X_0 was sent is actively misleading, not merely uninformative.

What the receiver actually acquires

AVERAGE OVER EVERY INPUT/OUTPUT PAIR, WEIGHTED BY ITS JOINT PROBABILITY

$$I(X,Y) = \sum_i \sum_j P(X_i,Y_j) \cdot \log_2 \left[P(X_i|Y_j) / P(X_i) \right] \quad \text{bits/symbol}$$

BAYES, THREE WAYS

$$P(X_i|Y_j) / P(X_i) = P(X_i,Y_j) / P(X_i)P(Y_j) = P(Y_j|X_i) / P(Y_j)$$

FROM THE TRANSMITTER'S SIDE

$$I(X,Y) = H(X) - H(X|Y)$$

H(X) sent − H(X|Y) equivocation

What reaches the receiver is what the source sent, minus what the channel destroyed.

FROM THE RECEIVER'S SIDE

$$I(X,Y) = H(Y) - H(Y|X)$$

H(Y) received − H(Y|X) error entropy

The same quantity: destination entropy minus the uncertainty injected purely by noise.

One physical quantity, viewed from opposite ends of the channel — and always I(X,Y) ≤ H(X), with equality only if the channel is noiseless.

How much of the source survives 2 % errors

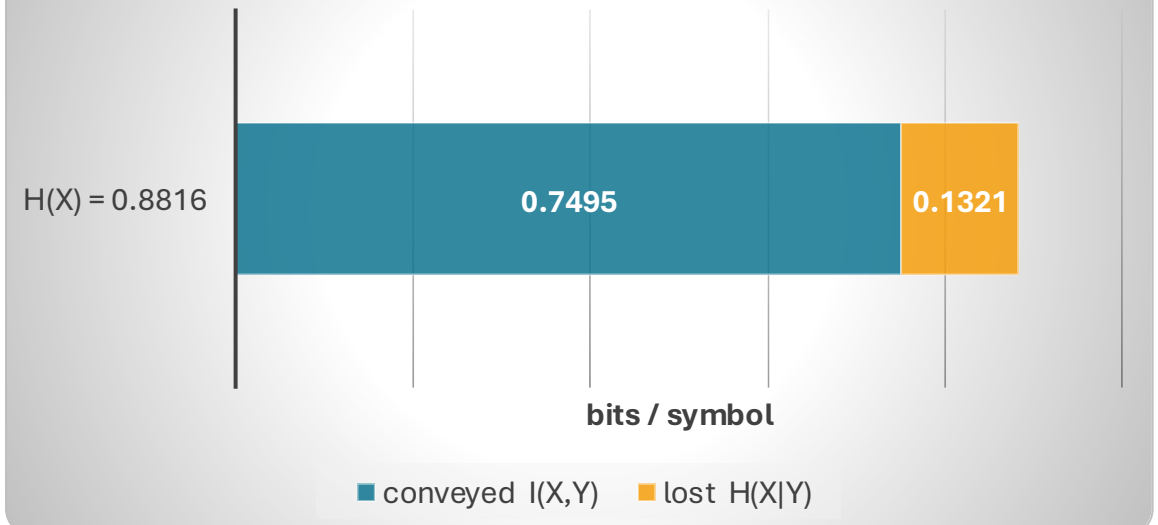
WEIGHT EACH PAIR BY ITS JOINT PROBABILITY

$$I(X,Y) = 0.294(1.670) + 0.686(0.502) + 0.006(-5.113) + 0.014(-3.945) \approx 0.7495$$

$$H(X) = 0.3(1.737) + 0.7(0.515) = 0.8816$$

$$H(X|Y) = H(X) - I(X,Y) = 0.8816 - 0.7495 \approx 0.1321$$

Where the source entropy goes



Source entropy
bits/symbol sent
$H(X) = 0.8816$

Conveyed
bits/symbol the receiver acquires
$I(X,Y) = 0.7495$

Lost in the channel
bits/symbol destroyed by noise
$H(X|Y) = 0.1321$

SUMMARY

Key formulas from Lecture 5

Pair mutual information

$$I(X_i,Y_j) = \log_2 [P(X_i|Y_j)/P(X_i)]$$

positive for correct pairs, negative for errors

Average mutual information

$$I(X,Y) = \sum\sum P(X_i,Y_j) I(X_i,Y_j)$$

the true bits/symbol the receiver acquires

Conveyed = sent − lost

$$I(X,Y) = H(X) - H(X|Y)$$

H(X|Y) is the equivocation

Conveyed = received − noise

$$I(X,Y) = H(Y) - H(Y|X)$$

H(Y|X) is the error entropy

The BSC

$$P(Y_1|X_0) = P(Y_0|X_1) = p_e$$

one crossover probability describes it all

The bound

$$I(X,Y) \le H(X)$$

equality only for a noiseless channel

NEXT: LECTURE 6 — CHANNEL CAPACITY

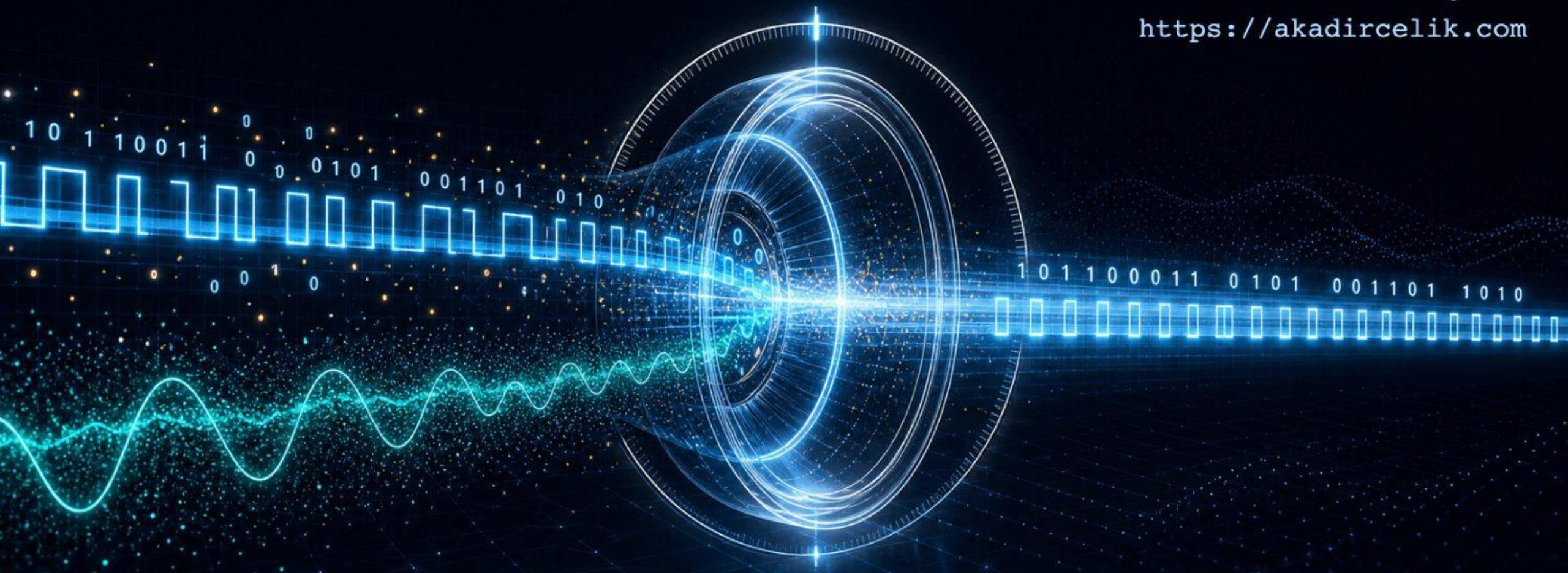
Maximise I(X,Y) over every possible source distribution and you get the channel capacity — the fastest rate at which information can cross the channel error-free.

ELEC3203 · DIGITAL CODING & TRANSMISSION

LECTURE 6

CHANNEL CAPACITY

WINTEL
Wireless Intelligence & Networked Sensing Laboratory
Prof. Abdulkadir Çelik
https://akadircelik.com



Lecture Notes
Podcast
Table of Content

Roadmap

Channel capacity · Shannon's theorem · the Gaussian channel · bandwidth vs SNR

01 **From mutual information to capacity**

Maximise I(X,Y) over every possible input distribution

02 **Discrete channels**

The noise-free bound, and the capacity of the BSC

03 **Shannon's channel coding theorem**

$R \leq C$ makes arbitrarily reliable transmission possible

04 **Analogue sources**

Differential entropy, and why Gaussian maximises it

05 **The Shannon-Hartley law**

$C = B \log_2(1 + S/N)$, and trading bandwidth against power

NOTATION

New symbols in this lecture

C	channel capacity, in bits/symbol or bits/second
t_{av}, T_s	average symbol duration, and the constant symbol duration
p_e	the BSC crossover (error) probability
B	channel bandwidth, in Hz
S_p, N_p	signal power and noise power
N_0	two-sided noise power spectral density; $N_p = N_0 B$
$p(x)$	probability density function of a continuous source
$H(x)$	differential entropy of a continuous source
σ^2_x	variance of a Gaussian source — equal to its power S_p

The best this channel could ever do

Different source distributions on the same channel give different I(X,Y). Capacity takes the maximum over all of them.

$$C = \max \{ I(X,Y) \}$$

bits / symbol

$$C = \max \{ I(X,Y) / t_{av} \}$$

bits / second, with $t_{av} = \sum P(X_i)\, t_i$

TWO CONDITIONS JOINTLY MAXIMISE I(X,Y)

A property of the channel

The channel must introduce no errors, H(Y|X) = 0.

Not something we control — it is set by the physics.

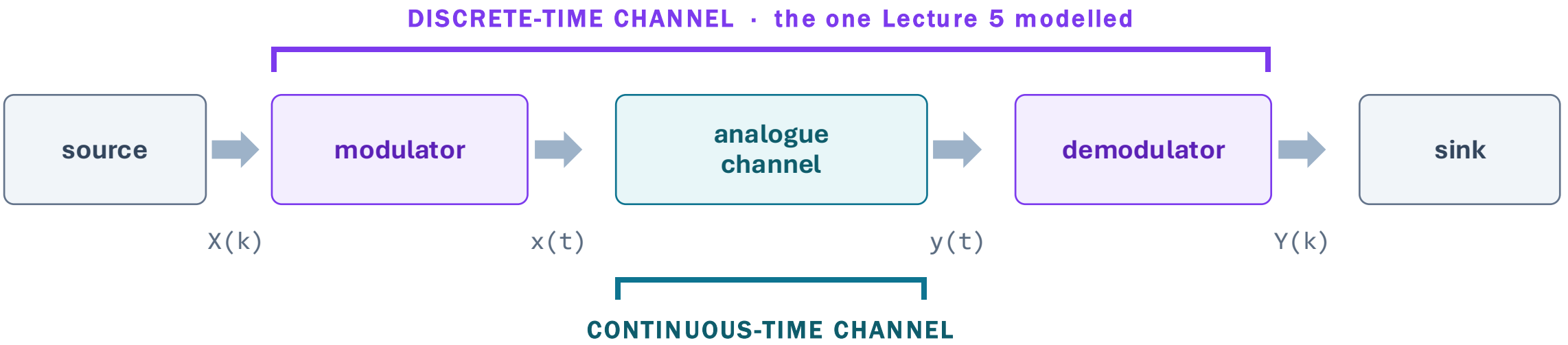
A property of the source

The inputs must be equiprobable.

With constant symbol durations, this is the maximum-entropy condition from Lecture 2.

Capacity is a property of the channel alone — achieved only when the source happens to be the right one.

Two channels, one inside the other



What the MODEM does

Turns discrete symbols X(k) into a continuous waveform x(t), and turns the received y(t) back into symbols Y(k).

Why it matters

Mutual information is measured across the discrete-time channel — which wraps the MODEM around the physical medium.

Capacity can be quoted for either one — bits per symbol for the discrete channel, bits per second once you include timing.

The upper bound with no errors at all

1 **No errors**

$$H(Y \mid X) = 0$$

the error entropy vanishes

2 **So all of it gets through**

$$I(X,Y) = H(Y) = H(X)$$

the channel conveys exactly the source entropy

3 **Per second**

$$C = \max \{ H(X) / t_{av} \}$$

divide by the average symbol duration

NOW FIX THE TIMING AND THE DISTRIBUTION

constant durations $t_i = T_s$ + equiprobable q-ary source $\Rightarrow$ $H(X) = \log_2 q$

$$C = \log_2 q / T_s \quad \text{bits / second}$$

The maximum achievable transmission rate over a noise-free channel — set purely by the alphabet size and the symbol duration.

Four steps to a one-parameter answer

1 Equiprobable inputs

$P(X_0) = P(X_1) = ½$, and by symmetry $P(Y_0) = P(Y_1) = ½$.

$$P(X_0,Y_0) = P(X_1,Y_1) = (1-p_e)/2$$
$$P(X_0,Y_1) = P(X_1,Y_0) = p_e/2$$

2 Average over the four pairs

Using Bayes in the form $P(X_i|Y_j)/P(X_i) = P(Y_j|X_i)/P(Y_j)$.

$$I(X,Y) = \Sigma_i\Sigma_j\ P(X_i,Y_j)\ \log_2\ [\ P(Y_j|X_i)\ /\ P(Y_j)\ ]$$

3 Substitute $P(Y_j) = ½$

Every ratio becomes $2\,P(Y_j|X_i)$.

two terms in $\log_2[2(1-p_e)]$, two terms in $\log_2(2p_e)$

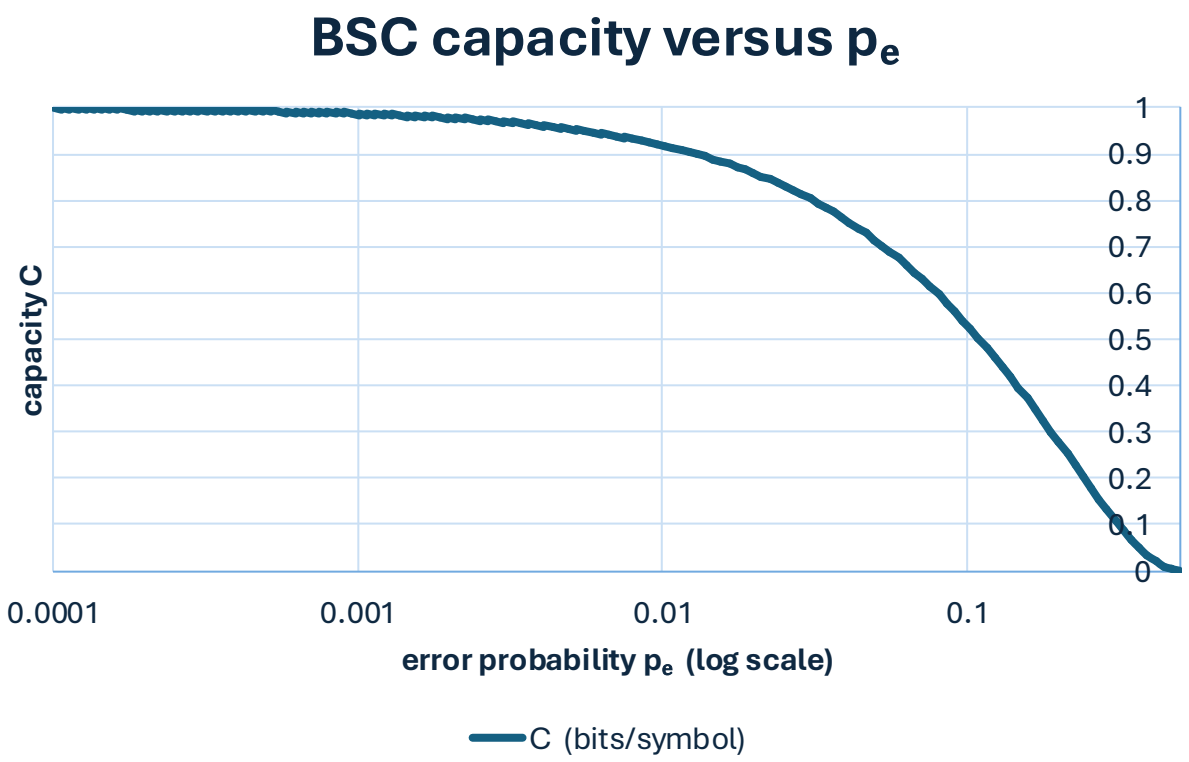
4 Collect and expand

Using $\log_2(2x) = 1 + \log_2 x$, the halves combine and the 1 factors out.

$$C = 1 + (1-p_e)\ \log_2(1-p_e) + p_e\ \log_2\ p_e$$

Capacity depends on the crossover probability p_e alone — nothing else about the channel matters.

How fast capacity falls away



Perfect channel

$$p_e = 0 \quad \Rightarrow \quad C = 1$$

the most a binary alphabet can carry

Useless channel

$$p_e = 0.5 \quad \Rightarrow \quad C = 0$$

output independent of input — nothing crosses

A 1 % error rate

$$p_e = 10^{-2} \quad \Rightarrow \quad C \approx 0.92$$

already costs about 8 % of capacity

$$C = 1 + (1 - p_e)\log_2(1 - p_e) + p_e \log_2 p_e \qquad \text{bits / symbol}$$

Noise limits the rate, not the reliability

THE THEOREM

If the information rate satisfies $R \leq C$, there exists a channel coding technique that transmits over the channel with arbitrarily small error probability. If $R > C$, error-free transmission is impossible.

$$R \leq C \Rightarrow \text{reliable coding exists}$$

$$R > C \Rightarrow \text{impossible, at any cost}$$

What it does say

A noisy channel is not an obstacle to reliable communication — only a limit on the rate at which it can happen.

What it does not say

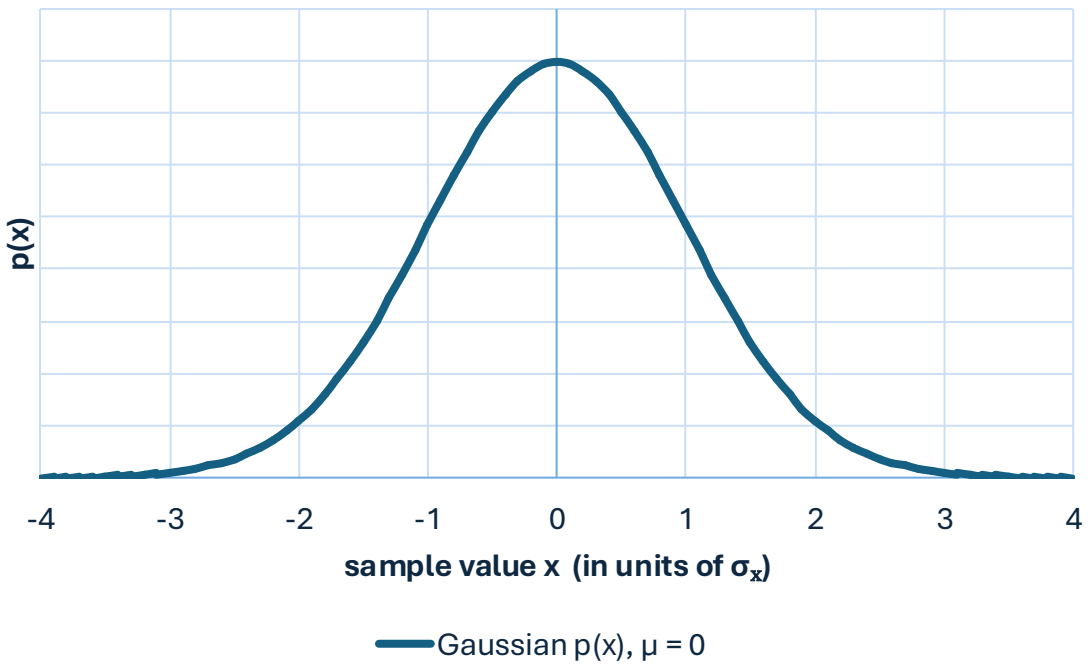
It guarantees a good code exists, but gives no recipe for constructing one.

Where practice stands

Most codes fall well short of C, but turbo and LDPC codes, with iterative detection-decoding, get close.

Differential entropy, and why Gaussian wins

The maximum-entropy density



Differential entropy

$$H(x) = -\int p(x) \log_2 p(x)\, dx$$

the continuous counterpart of source entropy

The Gaussian density

$$p(x) = (1/\sqrt{(2\pi)}\sigma_x)\, e^{(-x^2/2\sigma^2_x)}$$

zero-mean, with variance equal to the signal power

Its maximum entropy

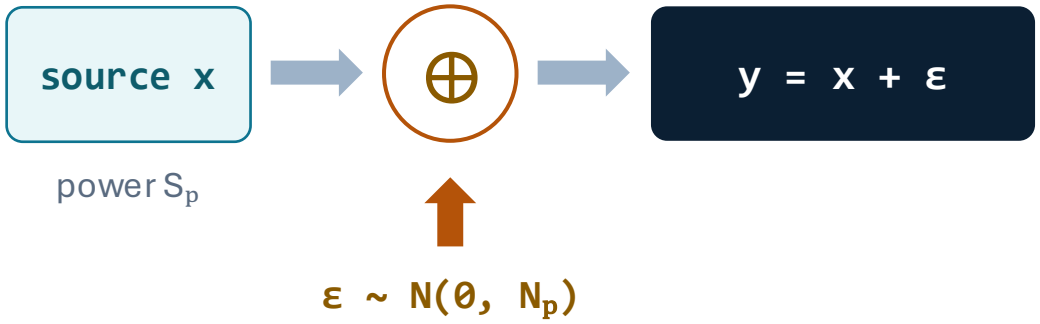
$$H_{max}(x) = \tfrac{1}{2} \log_2(2\pi e \sigma^2_x)$$

no other density of the same power beats it

An equiprobable discrete source maximises entropy; the Gaussian maximises differential entropy at a given power.

Gaussian in, Gaussian noise, Gaussian out

THE MODEL



THE TWO ENTROPIES SUBTRACT

$$\sigma^2_\gamma = S_p + N_p$$

$$H_{max}(y) = \tfrac{1}{2} \log_2(2\pi e(S_p + N_p))$$

$$H(y|x) = H(\varepsilon) = \tfrac{1}{2} \log_2(2\pi e\, N_p)$$

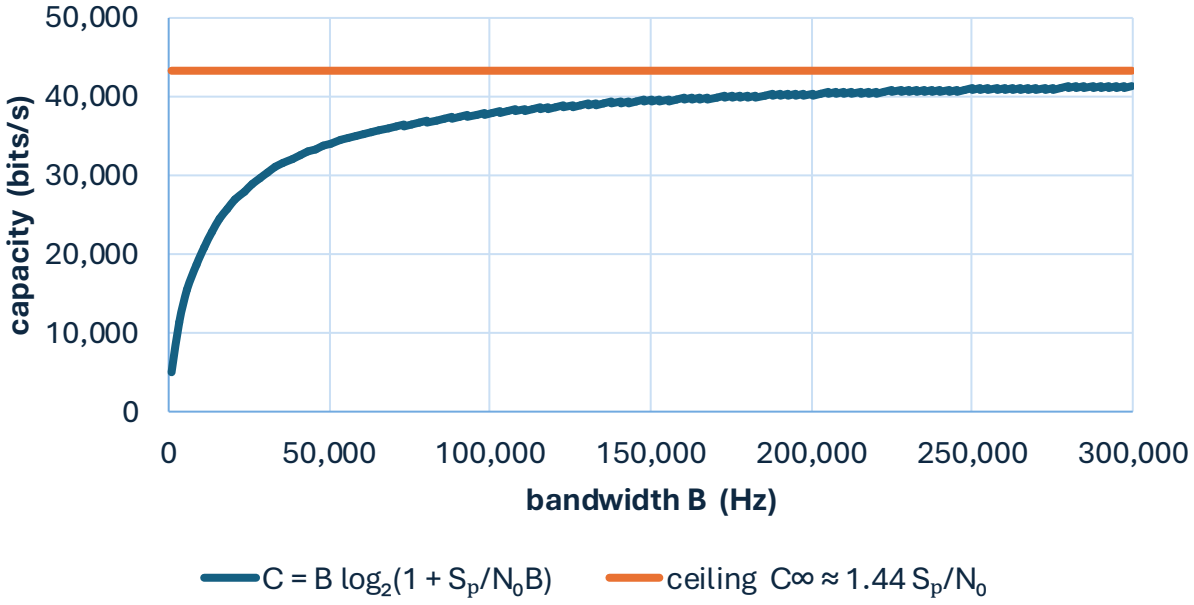
$$I(x,y) = H(y) - H(y|x)$$

$$I(x, y) = \tfrac{1}{2} \log_2 \left(1 + S_p / N_p \right) \quad \text{bits / symbol}$$

The 2πe factors cancel, leaving capacity as a function of the signal-to-noise ratio alone.

Bandwidth and power, traded against each other

Capacity versus bandwidth, fixed S_p/N_0



$$C = B \log_2(1 + S_p/N_p)$$

bits / second, with $N_p = N_0 B$

Only two resources

Bandwidth B and signal power S_p, through the SNR — either can be traded for the other at fixed C.

Diminishing returns

Capacity grows linearly with bandwidth but only logarithmically with power.

The infinite-bandwidth ceiling

$C_\infty = (S_p/N_0) \log_2 e \approx 1.44\, S_p/N_0$ — more band does not buy unlimited capacity.

A noiseless channel would have infinite capacity — but with real noise, capacity saturates however wide the band gets.

6.5 · WORKED EXAMPLE

A voice-grade telephone line

$$\text{Bandwidth } B = 3 \text{ kHz}, \quad \text{signal-to-noise ratio } S_p/N_p = 1000 \quad (30 \text{ dB})$$

Apply Shannon-Hartley

$$C = 3000 \cdot \log_2(1 + 1000)$$

Evaluate the logarithm

$$\log_2(1001) \approx 9.968$$

Capacity

$$C \approx 29{,}900 \text{ bits / second}$$

The ceiling behind “56k” modems: about 30–33 kbps is all that line can carry.

TRADING BANDWIDTH FOR POWER

Take a channel with SNR = 15 and halve its bandwidth. To hold C fixed, the signal power must rise by a factor of 8.5.

$$\text{half the band} \rightarrow 8.5\times \text{ the power}$$

WHY THE TRADE IS SO LOPSIDED

C rises linearly with bandwidth but only logarithmically with power. Doubling the band roughly doubles capacity; doubling the power adds barely one bit per symbol.

SUMMARY

Key formulas from Lecture 6

Channel capacity

$$C = \max \{ I(X,Y) \}$$

maximised over every input distribution

Noise-free channel

$$C = \log_2 q / T_s$$

bits/second, equiprobable q-ary source

BSC capacity

$$C = 1 + (1-p_e)\log_2(1-p_e) + p_e \log_2 p_e$$

depends on p_e alone

Shannon's theorem

$$R \leq C$$

arbitrarily reliable coding exists

Gaussian channel

$$I = \tfrac{1}{2} \log_2(1 + S_p/N_p)$$

bits/symbol, from differential entropy

Shannon-Hartley

$$C = B \log_2(1 + S_p/N_p)$$

bits/second, after Nyquist sampling

NEXT: LECTURE 7 — WRAP-UP AND TUTORIAL

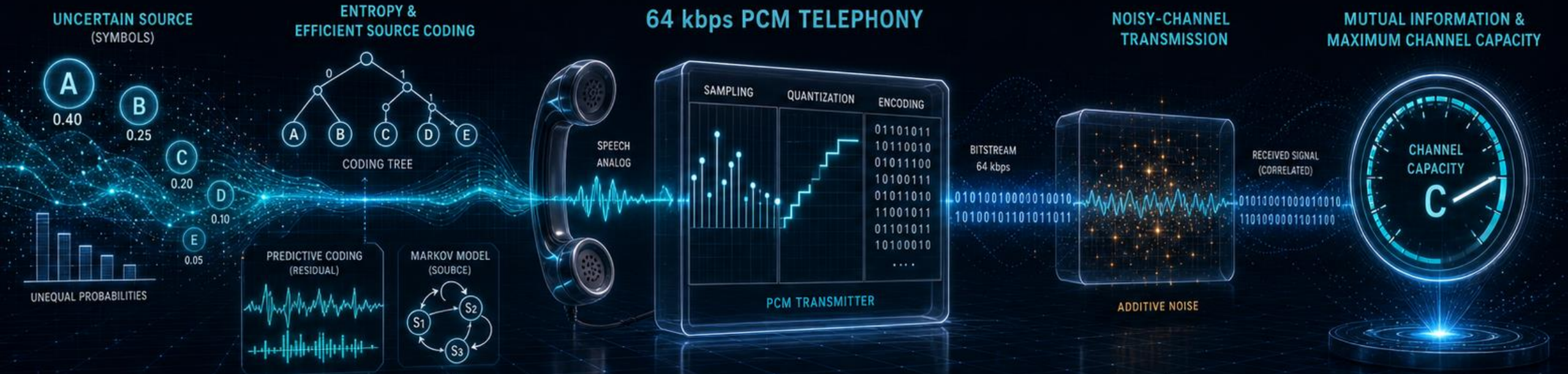
Whole-module revision across the source and channel sides, the 1960s 64 kbps speech-codec case study, and four worked exam-style problems.

ELEC 3203 · DIGITAL CODING & TRANSMISSION

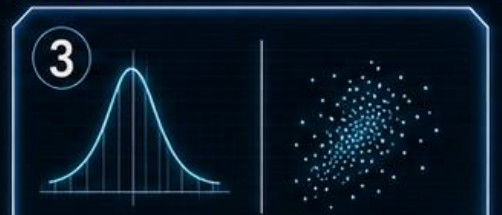
LECTURE 7

WRAP-UP AND TUTORIAL

WINTEL
Wireless Intelligence & Networked Sensing Laboratory
Prof. Abdulkadir Çelik
https://akadircelik.com



4 WORKED PROBLEMS



ELEC3203 · LECTURE 7

Roadmap

Whole-module revision · a real codec · four worked exam-style problems

01 **Revision — the source side**

Entropy, information rate and coding efficiency in one table

02 **Practical source coding**

Decorrelate first, then entropy-code the residual

03 **Revision — the channel side**

Mutual information and its two decompositions

04 **Capacity at a glance**

Noise-free, BSC and Gaussian channels side by side

05 **Case study and tutorial**

The 1960s 64 kbps speech codec, then Examples 7.1 to 7.4

7.1 · REVISION — THE SOURCE SIDE

Everything the source side asks of you

A digital source is fully specified by four things: its symbol set, the symbol probabilities, its symbol rate, and any interdependency.

quantity	formula	meaning
Information content	$I(m_i) = -\log_2 p_i$	bits carried by one occurrence of m_i
Entropy · memoryless	$H = -\sum p_i \log_2 p_i$	average information per symbol
Entropy · 1st-order Markov	$H = \sum_i P_i H_i$	state-averaged entropy, with memory
Information rate	$R = H \cdot R_s$	bits/s the source really needs
Coding efficiency	$\eta = R / R^b = H / L$	how close the coded rate gets to R

Efficient source coding aims to bring the coded bit rate R^b as close as possible to the information rate R.

7.2 · PRACTICAL SOURCE CODING

Always the same two steps

Entropy coding is optimal only for a memoryless source — and real signals never are.

STEP 1 · DECORRELATE

Remove the predictable part with a model, typically a predictor. What is left — the residual — is nearly white, so almost memoryless.

STEP 2 · ENTROPY-CODE

Apply Shannon-Fano or Huffman to that residual, or run-length coding if it is binary and mostly zeros. Now entropy coding is near-optimal.

Speech

Strong temporal correlation between successive samples — a linear predictor removes most of it.

Video

Spatial (intra-frame) correlation on top of temporal (inter-frame). Send one reference frame, then only the differences.

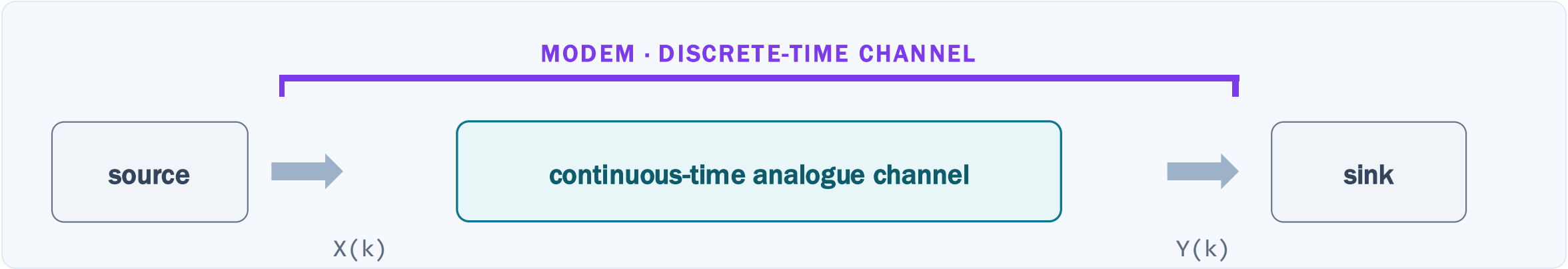
The tell

Whenever you meet a practical codec, look for these two steps — they are almost always there.

Frame differences are themselves often run-length coded — step 2 applied to the output of step 1.

7.3 · REVISION — THE CHANNEL SIDE

One channel, seen two ways



$$I(X,Y) = H(X) - H(X|Y)$$

sent − lost in the channel

$$I(X,Y) = H(Y) - H(Y|X)$$

received − injected by noise

WHAT MUTUAL INFORMATION MEASURES

The average information the receiver actually acquires per symbol — the same physical quantity whether you read it from the transmitter's end or the receiver's end.

7.4 · CHANNEL CAPACITY AT A GLANCE

Four channels, four formulas

channel	capacity	notes
Discrete, general	$C = \max \{ I(X,Y) \}$	maximised over input distributions; also C/t_{av} in bits/s
Noise-free q-ary	$C = \log_2 q / T_s$	equiprobable symbols of constant duration T_s
Binary symmetric	$C = 1 + (1-p_e)\log_2(1-p_e) + p_e \log_2 p_e$	$C = 1$ at $p_e = 0$; $C = 0$ at $p_e = 0.5$
Gaussian · Shannon-Hartley	$C = B \log_2(1 + S_p/N_p)$	$N_p = N_0 B$; trade bandwidth B against power S_p

SHANNON'S THEOREM, IN ONE LINE

If $R \leq C$ a coding scheme exists that transmits with arbitrarily small error. If $R > C$, error-free transmission is impossible — no matter how clever the code.

7.5 · CASE STUDY

The 1960s 64 kbps telephone speech codec

When the analogue telephone network was digitised, speech was coded at a fixed 64 kbps.

1 **Band-limit**
low-pass filter
`3.4 kHz`

2 **Sample**
at the Nyquist rate
`8 kHz`

3 **Compress**
μ-law companding
`non-linear`

4 **Quantise**
8 bits per sample
`256 levels`

64 kbps

THE ARITHMETIC

$$R^b = \log_2 256 \times 8\ \text{k} = 8\ \text{bits} \times 8\ \text{ksymbols/s} = 64\ \text{kbps}$$

WHY μ-LAW IS THERE

The compressor expands small-magnitude samples and compresses large ones, so the 256 quantised levels come out roughly equiprobable — it is itself a practical form of entropy coding.

7.5 · YOUR CRITICISM, AS AN INFORMATION THEORIST

One wrong answer, one right one

TEMPTING, BUT WRONG

“The samples are not equiprobable, so fixed 8-bit coding wastes bits — use entropy coding instead.”

Not so. The μ-law compressor has already flattened the distribution, so the 256 levels are close to equiprobable. Fixed 8-bit coding is near-optimal for what the quantiser produces.

THE REAL FLAW

The scheme treats speech as memoryless — but speech samples are strongly correlated.

So 64 kbps sits far above the true information rate. Remove the redundancy first with a predictive model, code the near-white residual, and the rate collapses.

THE SAME TWO STEPS AS SECTION 7.2

`decorrelate → entropy-code the residual`

A modern mobile speech codec carries the same speech at only a few kbps — an order of magnitude below the 1960s design.

TUTORIAL · EXAMPLE 7.1 (a)

Shannon-Fano on a six-symbol source

Six symbols in BCD at R_s = 9.6 kbaud: p(A) = 0.30, p(B) = 0.10, p(C) = 0.02, p(D) = 0.15, p(E) = 0.40, p(F) = 0.03

symbol	p_i	BCD	Shannon-Fano	bits
E	0.40	100	0	1
A	0.30	000	10	2
D	0.15	011	110	3
B	0.10	001	1110	4
F	0.03	101	11110	5
C	0.02	010	11111	5

Sorted by descending probability; split to balance the mass at every step.

Source entropy

$H = 2.0572$ bits/symbol

Information rate

$R = 9.6k \times 2.0572 = 19.75$ kbit/s

BCD data rate

$R^b = 3 \times 9.6k = 28.8$ kbit/s

Efficiency before coding

$\eta = 2.0572 / 3 = 68.6\ \%$

The disadvantage: codewords run from 1 to 5 bits

TUTORIAL · EXAMPLE 7.1 (b)

Decoding, rates, and Huffman compared

DECODE 11001111000011010110O

110	0	11110	0	0	0	110	10	110	0
D	E	F	E	E	E	D	A	D	E

Average codeword length

$L = 2.10$ bits/symbol

Data rate after coding

$9.6\text{k} \times 2.10 = 20.16$ kbit/s

Compression factor

$28.8 / 20.16 = 1.43\times$

Efficiency after coding

$\eta = 2.0572 / 2.10 = 98.0\ \%$

AND WITH HUFFMAN?

Merging bottom-up gives lengths 1, 2, 3, 4, 5, 5 — the same distribution as Shannon-Fano. So L = 2.10, the rate is 20.16 kbit/s and the efficiency is 98.0 % again; only the individual bit patterns differ.

TUTORIAL · EXAMPLE 7.2

Mutual information on a BSC

$$P(X_0) = p = 1/4, \quad P(X_1) = 3/4, \quad \text{crossover probability } p_e = 1/10$$

joint	Y_0	Y_1	$P(X_i)$
sent X_0	0.225	0.025	0.250
sent X_1	0.075	0.675	0.750
$P(Y_j)$	0.300	0.700	1.000

REVERSE CONDITIONALS, BY BAYES

$$P(X_0|Y_0) = 0.750 \qquad P(X_1|Y_0) = 0.250$$

$$P(X_0|Y_1) = 0.036 \qquad P(X_1|Y_1) = 0.964$$

$H(X|Y) \neq H(Y|X)$ in general — but both give the same $I(X,Y)$.

Source entropy
$H(X) = 0.8113$

Destination entropy
$H(Y) = 0.8813$

Error entropy
$H(Y|X) = 0.4690$

Conveyed
$I(X,Y) = 0.4123$

Equivocation
$H(X|Y) = 0.3990$

TUTORIAL · EXAMPLE 7.3

Markov source over a Gaussian channel

4-ary 1st-order Markov, $P = (0.2, 0.3, 0.2, 0.3)$, $R_s = 10^8$ symbols/s, AWGN with SNR = 63

from	to X_1	to X_2	to X_3	to X_4	H_i
X_1	0.6	0.1	0.1	0.2	1.5710
X_2	0.1	0.6	0.1	0.2	1.5710
X_3	0.1	0.1	0.8	0.0	0.9219
X_4	0.0	0.1	0.1	0.8	0.9219

Teal = sticky self-transitions; each row entropy from $H_i = -\sum p_{ij} \log_2 p_{ij}$.

Source entropy
$H = \sum P_i H_i = 1.2464$ bits/symbol

Information rate
$R = 10^8 \times 1.2464 \approx 124.6$ Mbit/s

Capacity available
$C = B \log_2(1 + 63) = 6B$

Minimum bandwidth
$B \geq R / 6 \approx 20.8$ MHz

WHY THE MEMORY MATTERS HERE

Ignoring the memory would give H(1) = 1.9710 bits/symbol and demand about 32.8 MHz. Accounting for the correlation cuts the required bandwidth by roughly a third.

TUTORIAL · EXAMPLE 7.4

Run-length coding a predictive residual

Residual at 3.1844 Mbit/s with $P(0) = p = 0.95$, RLC codeword length $n = 5$, so $N = 2^5 - 1 = 31$

Average input length

$d = (1 - 0.95^{31})/0.05 = 15.922$ bits

Compression ratio

$C = d / n = 15.922 / 5 = 3.1844$

Rate after RLC

$3.1844\ M / 3.1844 = 1.000$ Mbit/s

INPUT PATTERNS FOR EACH OUTPUT

00000	1	run of 0
00001	01	run of 1
00010	001	run of 2
⋮	⋮	⋮
11110	0…01 (31 bits)	run of 30
11111	0…00 (31 bits, no 1)	maximal run

DECODE 11011000000111110

11011	27 zeros then a 1	28 bits	00000	a single 1	1 bit	11110	30 zeros then a 1	31 bits

ELEC3203 · INFORMATION THEORY

The module in one line

Measure the information, squeeze out the redundancy, then push it across the channel as fast as capacity allows.

THE SOURCE SIDE

Information content, entropy and information rate say how much a source truly produces. Shannon-Fano, Huffman and run-length coding bring the transmitted bit rate down towards it — after a predictor has removed any memory.

THE CHANNEL SIDE

Mutual information says how much survives the noise; capacity is its maximum over all inputs. Shannon's theorem promises reliable coding below capacity, and Shannon-Hartley prices it in bandwidth and power.

$H = -\sum p \log_2 p$

$R = R_s H$

$\eta = H / L$

$I = H(X) - H(X|Y)$

$C = B \log_2(1 + S/N)$